ibai-publishing
Prof. Dr. Petra Perner
PF 30 11 38
04251 Leipzig, Germany
E-mail: info@ibai-publishing.org

P-ISSN 1864-9734
E-ISSN 2699-5220
ISBN 978-3-942952-93-4

www.ibai-publishing.org

ISBN 978-3-942952-93-4

Machine Learning and Data Mining in Pattern Recognition, Proceedings, MLDM 2022

Petra Perner

Petra Perner (Ed.)

Machine Learning and Data Mining in Pattern Recognition

18th International Conference on Machine Learning and Data Mining, MLDM 2022

New York, USA, July 16-21, 2022

Proceedings

ibai - publishing

Petra Perner (Ed.)

Machine Learning and Data Mining in Pattern Recognition

18th International Conference on Machine Learning and Data Mining, MLDM 2022, New York, USA, July 16-21, 2022
Proceedings

ibai Publishing
www.ibai-publishing.org

Volume Editor

Petra Perner
Institute of Computer Vision and Applied Computer Sciences IBaI
Hertha-Lindner-Str. 10-12
01067 Dresden
E-mail: pperner@ibai-institut.de

The German National Library listed this publication in the German National Bibliography.
Detailed bibliographical data can be downloaded from http://dnb.ddb.de.

ibai-publishing
Prof. Dr. Petra Perner
PF 30 11 38
04251 Leipzig, Germany
E-mail: info@ibai-publishing.org
http://www.ibai-publishing.org

P-ISSN 1864-9734
E-ISSN 2699-5220
ISBN 978-3-942952-93-4

Editorial

The eighteenth event of the International Conference on Machine Learning and Data Mining MLDM was held in New York (www.mldm.de) running under the umbrella of the World Congress "The Frontiers in Intelligent Data and Signal Analysis, DSA2022" (www.worldcongressdsa.com).

At a time when we are still struggling with the corona pandemic, we scientists from different nations have gathered together for a peaceful discourse on an important research focus in the field of data mining and machine learning.

With our conference, we scientists show that we respect the opinions and work of others. That we are ready to consider them peacefully and in friendship under the critical view of the high scientific standards that this conference has.

The International Program Committee has done an excellent and time-consuming job to select the best papers and provide important guidance on the work of the authors. I would like to thank all the members of the Program Committee for their efforts and that you have contributed with your top-class scientific competence.

The best papers are presented at this conference. The acceptance rate is 33%.

Thank you to all the scientists who have participated in this conference with your excellent work.

A special issue will be done after the conference in the Intern. Journal Transactions on Machine Learning and Data Mining (http://www.ibai-publishing.org/journal/mldm/about.php).

I would also like to thank those scientists who have participated in the conference with their work and have not been successful. Even if we have rejected work, we hope that the indications of the program committee will encourage you to reconsider your work and that you will perhaps face the critical scientific consideration of your work by the international program committee again next year.

The tutorial days rounded up the high quality of the conference. Researchers and practitioners got an excellent insight in the research and technology of the respective fields, the new trends and the open research problems that we like to study further.

A tutorial on Data Mining and a tutorial on Case-Based Reasoning, were held after the conference.

I also thank the members of the Institute of Computer Vision and applied Computer Sciences, Germany (www.ibai-institut.de), who handled the conference as secretariat. We appreciate the help and understanding of the editorial staff at ibai-publishing house, who supported the publication of these proceedings (http://www.ibai-publishing.org/html/proceeding.php).

Last, but not least, we wish to thank all the speakers and participants who contributed to the success of the conference. We hope to see you in 2023 in New York again at the next World Congress on "The Frontiers in Intelligent Data and Signal Analysis, DSA 2023" (www.worldcongressdsa.com), which combines under its roof the following three events: International Conferences Machine Learning and Data Mining, MLDM (www.mldm.de), the Industrial Conference on Data Mining, ICDM (www.data-mining-forum.de, and the International Conference on Mass Data Analy-

sis of Signals and Images in Medicine, Biotechnology, Chemistry, Biometry, Security, Agriculture, Drug Discovery and Food Industry, MDA (www.mda-signals.de), the workshops and tutorials.

July 2022 Petra Perner

17th International Conference on Machine Learning and Data Mining MLDM 2022

www.mldm.de

Program Chair

Petra Perner ..... Institute of Computer Vision and Applied Computer Sciences IBaI, Germany

Program Committee

Piotr Artiemjew	University of Warmia and Mazury in Olsztyn, Poland
Sung-Hyuk Cha	Pace Universtity, USA
Ming-Ching Chang	University of Albany, USA
Mark J. Embrechts	Rensselaer Polytechnic Institute and CardioMag Imaging, Inc, USA
Youssef Hadi	Ibn Tofail University, Marokko
Robert Haralick	City University of New York, USA
Naman Kapoor	Delhi Technological University, India
Dimitris Karras	Chalkis Institute of Technology, Greece
Adam Krzyzak.....	Concordia University, Canada
Chengjun Liu.....	New Jersey Institute of Technology, USA
Vincent Oria	New Jersey Institute of Technology, USA
Krzysztof Pancerz.	University Rzeszow, Poland
Dan Simovici.....	University of Massachusetts Boston, USA
Agnieszka Wosiak	Lodz University of Technology, Poland

Table of Content

Application of Bayesian STRIM to Datasets Generated via Partial Correspondence Hypothesis

Yuichi Kato[1] and Tetsuro Saeki[2]

[1] Shimane University,
1060 Nishikawatsu-cho, Matsue city, Shimane 690-8504, JAPAN
ykato@cis.shimane-u.ac.jp
[2] Yamaguchi University,
2-16-1 Tokiwadai, Ube city, Yamaguchi 755-8611, JAPAN
tsaeki@yamaguchi-u.ac.jp

Abstract. A statistical test rule induction method (STRIM) was proposed for an if-then rule induction method with a decision table dataset independently of rough sets theory while not utilizing the notion of approximation, with the validity of the method confirmed via a simulation experiment using a data generation (DG) model. Nevertheless, the previous DG model used a plain hypothesis of the complete correspondence with rules, and in this study, the model was improved using a hypothesis similar to human rating and the rule induction method for adaption to real-world datasets. Specifically, first, the hypothesis was expanded from a complete correspondence hypothesis to a partial correspondence hypothesis, and second, the previous STRIM was developed into a Bayesian STRIM that infers and/or explores the causes on the basis of the results. The validity and efficacy were confirmed by applying the Bayesian STRIM to a verification system, whereas the relationship and difference between the Bayesian STRIM and a maximum a posteriori probability estimate and a Bayesian network method were also studied in relation to the rule induction problem.

Keywords: rough sets · if-then rule induction · Bayesian STRIM · simulation experiment.

1 Introduction

As one of the various existing data mining methods, the rough sets (RS) method, which is used for inducing if-then rules hidden in a dataset known as a decision table (DT), has been proposed. A DT is a sample dataset pertaining to a population of interest that is arranged in terms of various attributes and the attendant values. These attributes are known as condition attributes (CAs), and their values are used for determining a category termed the decision attribute's (DA's) value, which represents the category or grade of the sample. Accordingly, once the DT has been provided as a learning dataset and if-then rules have been induced

by combining certain pairs of CAs with the value acting as the condition part of the rules, with the DA's value as the decision part, a small number of rules can be used to arrange a large number of datasets, that is, the induced rules clearly present the structure of the population and can provide useful knowledge on it. Zhang et al. summarized the conventional RS methods, including the original RS theory devised by Pawlak [1], after reviewing a total of 110 studies [2].

However, the conventional RS methods were based on set theory and the attendant logic, and lacked the statistical view that the DT is simply a sample dataset obtained from the population of interest so that they induced different rule sets with every DT obtained from the same population, that is, statistically significant rule sets could not be induced. Hence, the statistical test rule induction method (STRIM) and its attendant algorithm were proposed using a data generation (DG) model and a verification system of induced rules (VSofIR), which compensated for the shortfalls of the conventional RS methods [3–8]. Here, the DG model presents a simulation model for generating DTs that obey prespecified rules (details are provided later).

Meanwhile, in terms of practical use, the DG model should be matched to a specific circumstance. For example, when rating the grade of a given sample, a large number of raters rate the various attributes of the sample and determine the total grade on the basis of their rated attribute results and their own if-then rules, using the decision part of the rules if the condition part of the used rules is met. Nevertheless, in the case where the condition part is not met, the raters will use their rules while considering the degree of partial correspondence to their rules, which is what generally occurs in actuality. In this study, this rating process is labeled a partial corresponding hypothesis (PCH) to distinguish it from a complete corresponding hypothesis (CCH), and the corresponding rule induction method is proposed and investigated in the following terms:

(1) A new DG model incorporating PCH is developed using the conventional DG model, with a DT generated on the basis of PCH as the PCH dataset.
(2) The problems caused by the previous STRIM are clarified by applying it to the PCH dataset to obtain insights into improving the method.
(3) A new rule induction method termed Bayesian STRIM is proposed along with the attendant procedure in view of adapting the new DG model, with its validity and efficacy confirmed using a VSofIR.
(4) The relationship and difference between the Bayesian STRIM and a maximum a posteriori (MAP) probability estimate and a Bayesian network (BN) method are considered in relation to the rule induction problem.

2 Improvement of Previous DG Model

In statistics, a dataset $U = \{u(i)|i = 1, ..., N = |U|\}$ is collected from a population of interest to estimate and infer the properties and features of the population. Here, $u(i)$ is an object with several attributes, the properties and features of which contribute to the estimation and inference of the population. Let us denote an observation system labeled the DT in the RS theory by

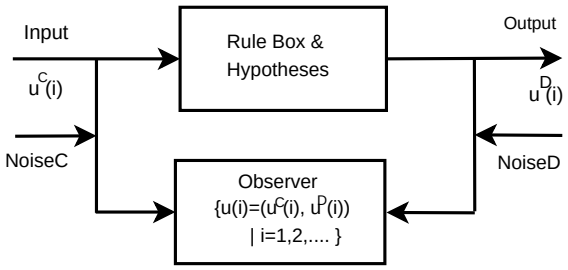


Fig. 1. Simulation model for DG and verification of induced rules. The rule box contains if-then rules $R(d,k)$: if $sCP(d,k)$ then $D=d$ $(d=1,2,...,k=12,...)$.

Table 1. Complete correspondence hypotheses with regard to the input.

Hypothesis 1	$u^C(i)$ coincides with $R(d,k)$, and $u^D(i)$ is uniquely determined as $D=d$ (uniquely determined data).
Hypothesis 2	$u^C(i)$ does not coincide with any $R(d,k)$, and $u^D(i)$ can only be determined randomly (indifferent data).
Hypothesis 3	$u^C(i)$ coincides with several $R(d,k)$ $(d=d1,d2,...)$, and their outputs of $u^C(i)$ conflict with each other. Accordingly, the output of $u^C(i)$ must be randomly determined from the conflicted outputs (conflicted data).

$S=(U,A=C\cup\{D\},V)$. Here, A is a set of an attribute, D is a DA and a response variable, and $C=\{C(j)|j=1,...,|C|\}$ is a set of the condition attribute $C(j)$ and a tuple of explanatory variables for the response variable. Meanwhile, V is the set of the attribute's values, that is, $V=\bigcup_{a\in A}V_a$ and V_a is the set of the value of attribute a. When randomly sampling $u(i)$ from the population, each attribute becomes a random variable with the respective attribute value as its outcome.

Figure 1 outlines the DG process [3, 4]. By randomly sampling $u(i)$ from the population, we obtain the outcome of $C=(C(1),...,C(|C|))$; that is, $u^C(i)=(v_{C(1)}(i),...,v_{C(|C|)}(i))$ is obtained and becomes the input into a rule box. The rule box transforms $u^C(i)$ into the output $u^D(i)$ using the rule box's prespecified rules $R(d,k)$: if $sCP(d,k)$ then $D=d$ $(d=1,2,...,k=1,2,...)$. Table 1 shows the subsequent PCH incorporating the input-modifying CCH.

The PCH estimates the degree Dgr of $u^C(i)$ corresponding to the box's prespecified rules and the rule of the highest Dgr is applied for transforming $u^C(i)$ into $u^D(i)$. If several rules of ties exist, one of them is randomly determined in the same way as hypothesis 3 shown in Table 1. The PCH expands and generalizes three cases for $u^C(i)$ (Table 1) while considering human decision making processes. Here, the observer shown in Fig. 1 records $u(i)=(u^C(i),u^D(i))$. Following this, NoiseC and NoiseD could be introduced to adapt the model for real-world datasets. Here, NoiseC adjusts the value of $u^C(i)=(v_{C(1)}(i),...,v_{C(|C|)}(i))$ or makes $v_{C(j)}(i)$ a missing value, whereas NoiseD adjusts the value of $u^D(i)$.

Table 2. Examples of prespecified rules in the rule box.

$R(d,k)$	$sCP(d,k)$	$D=d$	$(dpl, rdct)$
$R(1,1)$	110010	$D=1$	(0,0)
$R(1,2)$	001101		
$R(2,1)$	220020	$D=2$	(1,0)
$R(2,2)$	022202		
$R(3,1)$	330030	$D=3$	(0,1)
$R(3,2)$	003300		
$R(4,1)$	440040	$D=4$	(1,1)
$R(4,2)$	040404		
$R(5,1)$	550050	$D=5$	(1,2)
$R(5,2)$	050500		

On generating $u^C(i) = (v_{C(1)}(i), ..., v_{C(|C|)}(i))$ using random numbers and transforming it into $u^D(i)$ using the model shown in Fig. 1, including PCH, we obtain $U = \{u(i) = (u^C(i), u^D(i)) | i = 1, ..., N = |U|\}$ and subsequently apply it to any rule induction method to investigate the extent to which the method induces the prespecified rules. As such, the model (Fig. 1) can also be used as a verification system for the applied rule induction method.

3 Simulation Experiment applying the Previous STRIM to the PCH dataset and the Attendant Problems

We implemented the DG process with the PCH and the verification process to examine the ability of the applied method as follows:

(1) Various types of rules were specified (e.g., those shown in Table 2) and set in the rule box (Fig. 1), where $|C| = 6$, $V_a = \{1, 2, ..., 5\}$ $(a = C(j)(j = 1, ..., |C|), a = D)$, $sCP(1,1) = 110010$ denotes $sCP(1,1) = (C(1) = 1) \wedge (C(2) = 1) \wedge (C(5) = 1)$, before we labeled a rule of the rule length 3 $(RL = 3)$ with three conditions and the column of $(dpl, rdct)$ denoting the number of points of duplication and reduct, respectively. For example, two rules for $D = 5$ have one duplication point at $C(2) = 5$ and two reduct points at $C(3)$ and $C(6)$, which have nothing to do with those rules.

(2) Next, $v_{C(j)}(i)$ $(j = 1, ..., |C| = 6)$ was generated with a uniform distribution and $u^C(i) = (v_{C(1)}(i), ..., v_{C(6)}(i))$ $(i = 1, ..., N = 10,000)$ was formed.

(3) Then, $u^C(i)$ was transformed into $u^D(i)$ using the prespecified rules listed in Table 2 and the PCH, without generating NoiseC and NoiseD for a simple experiment. Here, the Dgr was estimated using the sum of the number of the conditions satisfied for each rule.

Table 3 shows various examples of $(u^C(i), R(d,k)) = Dgr$, which includes $(u^C(1) = 112233, R(1,1)) = 2$, $(u^C(1) = 112233, R(1,2)) = 0$, with $R(1,1)$ or $R(2,2)$ with the largest Dgr randomly used. If the CCP had been used, the rules with a larger RL would almost certainly not have been used. For

Table 3. Examples of $u^C(i)$ and its Dgr by prespecified rules.

$u^C(i) \setminus R(d,k)$	$R(1,1)$ 11001	$R(1,2)$ 00011	$R(2,1)$ 01220	$R(2,2)$ 02022	$R(3,1)$ 02330	$R(3,2)$ 03003	$R(4,1)$ 03300	$R(4,2)$ 04400	$R(5,1)$ 04040	$R(5,2)$: 04550
$u^C(1) = 112233$	2	0	0	2	1	0	0	0	0	0
$u^C(2) = 334455$	0	0	0	0	2	0	0	1	1	1
$u^C(3) = 123451$	1	1	1	1	0	1	0	1	1	0
$u^C(4) = 245512$	1	0	1	1	0	0	1	1	0	1
...	...	...	...	...	...	...	...	...	...	...

example, if $RL = 4$, the probability of using such rules would be approximately $(1/5)^4(4/5)^2 = 1.02E-3$. However, human beings will generally use one of the rules that better match their own rules. When $RL = 1$ is the largest Dgr, one among the large number of rules ($u^C(3)$ or $u^C(4)$ in Table 3) will be used. As such, the PCH complicates the rule induction problem compared with the CCH. The dataset generated on the basis of the above procedures will hereafter be referred to as the PCH dataset.

After randomly sampling $N_B = 5,000$ data from the generated dataset and forming a new dataset as the DT, we applied the previous STRIM (details in [7, 8]) to the PCH dataset. The previous method involves the basic idea that the rule condition part $CP = \bigwedge_j (C(j) = v_{j_k})$ causes the bias at $D = d$ in the distribution $f(n_1, n_2, ..., n_{|V_D|})$ of $D = m$, with this idea based on the following consideration and principle: the simultaneous probability of (C, D) can be denoted as $P(C, D) = P(C)P(D|C)$, where $P(D|C) = P(\text{if } C \text{ then } D)$. In the special case for $CP = \bigwedge_j (C(j) = v_{j_k})$ and $D = d$,

$$P(D = d|CP = \bigwedge_j (C(j) = v_{j_k})) = P(\text{if } CP = \bigwedge_j (C(j) = v_{j_k}) \text{ then } D = d). \tag{1}$$

Meanwhile, a CP satisfying

$$P(D = d|CP) = P(D = d) \tag{2}$$

cannot be a rule candidate.

To efficiently explore the CP, the previous STRIM first executes a statistical test specifying a null hypothesis H_0: $D = d$ that is independent of $CP = C(j)$ corresponding to Eq. (2) (the alternative H_1: $D = d$ is not independent of $CP = C(j)$). If H_0 is rejected, that is, H_1 is adopted, then $CP = (C(j) = v_{j_k})$, which is the dominant value for $D = d$ applied corresponding to Eq. (1). If H_0 is not rejected, $CP = C(j)$ is independent of $D = d$ corresponding to Eq. (2) and $CP = C(j)$ is reducted. Table 4 shows the results of this procedure, which is termed a reduct table (RT) [6]: $RT = \{rd(j, d) \in V_{C(j)} | j = 1, ..., |C|, d = 1, ..., |V_D|\}$, where r denotes "reduct." As the table shows, $C(j) = v_{j_k} = rd(j, d)$ exhibits a strong connection with $D = d$, and the if-then rule for $D = d$ can thus be constructed using a combination of both. For example, with $D = 2$, $C(j) = 2 = rd(j, 2)$ $(j = 1, ..., 6)$, which is in line with the data shown in Table 2.

Table 4. Reduct table for the DT generated based on the rules in Table 2 and the PCH: "r" means reduct.

$C(i) \setminus D = d$	$d = 1$	$d = 2$	$d = 3$	$d = 4$	$d = 5$
$C(1)$	1	2	3	4	5
$C(2)$	1	2	3	4	5
$C(3)$	1	2	3	r	r
$C(4)$	1	2	3	4	5
$C(5)$	1	2	3	4	5
$C(6)$	1	2	r	4	r

Table 5 shows representative results for the 41 rule candidates constructed through the combination of $RL = 1, 2$ and 3 for $D = 2$ in descending order of z-value [7, 8]. For example, the first row, $CndCP(2, 1)$, of the table denotes that the condition part of the induced rule candidate is $(C(2) = 2)$. Meanwhile, the frequency distribution of D: $f = (n_1, ..., n_5)$ satisfying the condition is $(238, 1, 346, 150, 176, 100)$, which suggests that the frequency exhibits a large bias of $n_{d=2} = 1,346$, and thus, $D = 2$ presents the decision part for the rule candidate. The distribution of $z = \frac{n_d + 0.5 - np_d}{(np_d(1-p_d))^{0.5}}$ obeys the standard normal distribution under the null hypothesis H_0: $CndCP$ is not a rule candidate (the alternative hypothesis H_1: $CndCP$ is a rule candidate) and the testing condition [9], $np_d \geq 5$ and $n(1 - p_d) \geq 5$, where $n = \sum_{m=1}^{5} n_m$. The p-value corresponding to the z-value is the index supporting H_0, with the accuracy and coverage also shown in Table 5. All the rule candidates in Table 5 exhibit partial rules of $R(2, 1)$ or $R(2, 2)$ prespecified in Table 2 or partially straddling rules between the two (e.g., $CndCP(2, 21)$ and $CndCP(2, 22)$).

The previous STRIM arranges the candidates using the p-values and the inclusion relationships among them. For example, $CndCP(2, 1)$ includes $CndCP(2, 2)$ as a special case, and its p-value is less than that of $CndCP(2, 2)$, whereas the same applies to $CndCP(2, 3)$ and so on. Thus, $CndCP(2, 1)$ can represent $CndCP(2, k)$ $(k = 2, 3, ..., 40)$. Applying the same to the rest, the previous STRIM, which was developed using the CCH dataset, induced the final results shown in Table 6. The previous STRIM induces the partial and/or straddle rules of the prespecified rules, as shown in Tables 5 and 6, which is natural since the PCH allows for prespecified rules to behave as their partial and/or straddle rules. However, the previous STRIM cannot effectively arrange these rules and subsequently induce original ones.

4 Improved STRIM Algorithm for the PCH Dataset

As noted above, the previous STRIM was based on Eqs. (1) and (2), and we used the bias in the distribution $f(n_1, n_2, ..., n_{V_D})$ of D for detecting Eq. (2) and experimentally studied its rule induction procedures and the attendant problems. Following this, we reviewed the principles as $P(CP, D = d) = P(D = d)P(CP|D = d)$ and the process of inducing the rules corresponding to Eqs. (1)

Table 5. Representative rule candidates for $D = 2$ induced by the previous STRIM for the PCH dataset.

$CndCP(d,k)$	$C(1)...C(6)$	D	p-value(z)	Accuracy	Coverage	$f = (n_1, n_2, ..., n_5)$
$CndCP(2,1)$	020000	2	0.00(41.2)	0.670	0.509	(238,1346,150,176,100)
$CndCP(2,2)$	022000	2	6.11E-211(31.0)	0.910	0.153	(16,406,2,16,6)
$CndCP(2,3)$	220000	2	3.17E-192(29.6)	0.915	0.138	(16,366,6,12,0)
$CndCP(2,4)$	020002	2	7.08E-191(29.4)	0.940	0.131	(8,346,10,2,2)
$CndCP(2,5)$	020200	2	5.80E-187(29.1)	0.890	0.141	(20,374,8,12,6)
$CndCP(2,6)$	200020	2	1.97E-170(27.8)	0.883	0.131	(20,346,6,8,12)
$CndCP(2,7)$	020020	2	4.25E-169(27.7)	0.902	0.125	(18,330,6,12,0)
$CndCP(2,8)$	000202	2	1.34E-160(27.0)	0.837	0.136	(16,360,18,14,22)
$CndCP(2,9)$	002200	2	3.57E-146(25.7)	0.833	0.125	(20,330,16,16,14)
$CndCP(2,10)$	002002	2	1.5E-126(23.9)	0.814	0.113	(14,298,20,20,14)
...	...	...	...	...	...	...
$CndCP(2,20)$	002020	2	1.25E-62(16.7)	0.648	0.089	(38,236,24,40,26)
$CndCP(2,21)$	220002	2	7.46E-58(16.0)	0.979	0.036	(2,94,0,0,0)
$CndCP(2,22)$	200022	2	1.72E-57(15.9)	1.000	0.034	(0,90,0,0,0)
$CndCP(2,23)$	020202	2	1.72E-57(15.9)	1.000	0.034	(0,90,0,0,0)
...	...	...	...	...	...	...
$CndCP(2,33)$	002220	2	4.05E-42(13.5)	0.889	0.030	(4,80,2,2,2)
$CndCP(2,34)$	220200	2	1.01E-41(13.5)	1.000	0.024	(0,64,0,0,0)
$CndCP(2,35)$	220020	2	1.01E-41(13.5)	1.000	0.024	(0,64,0,0,0)
...	...	...	...	...	...	...
$CndCP(2,40)$	020220	2	1.55E-36(12.6)	0.939	0.023	(4,62,0,0,0)
$CndCP(2,41)$	202002	2	3.37E-24(10.1)	0.813	0.020	(0,52,4,4,4)

and (2) as follows:

$$P(CP|D = d) = P(\text{if } D = d \text{ is given, the cause is } CP). \tag{3}$$

Meanwhile, a CP satisfying

$$P(CP|D = d) = P(CP) \tag{4}$$

cannot be a rule candidate. Equation (3) indicates the process for exploring the cause CP from the result $D = d$ (see Fig. 1), with this process not using U to explore the CP but only $U(d) = \{u(i)|u^{D=d}(i)\}$. In the same way as Eqs. (1) and (2), the CP in Eq. (3) will be constructed by the elements $C(j) = v_{j_k}$

Table 6. Final results for $D = 2$ induced by the previous STRIM for the PCH dataset.

$CP(d,k)$	$C(1)...C(6)$	D	p-value(z)	Accuracy	Coverage	$f = (n_1, n_2, ..., n_5)$
$CP(2,1)$	020000	2	0.00(41.2)	0.670	0.509	(238,1346,150,176,100)
$CP(2,2)$	200020	2	1.97E-170(27.8)	0.883	0.131	(20,346,6,8,12)
$CP(2,3)$	000202	2	1.34E-160(27.0)	0.837	0.136	(16,360,18,14,22)
$CP(2,4)$	002200	2	3.57E-146(25.7)	0.833	0.125	(20,330,16,16,14)
$CP(2,5)$	002002	2	1.50E-126(23.9)	0.814	0.113	(14,298,20,20,14)

Table 7. Examples of $CTT(D = 2)$ for the PCH dataset in Table 5 ("-" denotes $C(j) \neq rd(j,d)$, $|U(d=2)| = 1,323$, $p0 = 1.0E-5$).

Combination No. $Cmb(k)$	(d, j_1, j_2)	$freq.(d, j_1, j_2)$	$P(d, j1, j2)$ under H_0	p-value	$freq.(p0)$
Cmb(1)	- 2 2 - - -	82	0.0171	3.75E-23	45
Cmb(2)	- 2 - 2 - -	75	0.0168	1.14E-19	45
Cmb(3)	- 2 - - 2 -	72	0.0163	1.04E-18	44
Cmb(4)	2 2 - - - -	69	0.0159	1.51E-17	43
Cmb(5)	2 - - - 2 -	64	0.0155	1.98E-15	42
Cmb(6)	- - - 2 - 2	61	0.0163	5.37E-13	44
Cmb(7)	- - 2 - - 2	57	0.0166	8.51E-11	44
Cmb(8)	- 2 - - - 2	50	0.0162	3.13E-08	44
Cmb(9)	- - 2 2 - -	51	0.0172	7.41E-08	45
Cmb(10)	- - - 2 2 -	26	0.0163	1.45E-01	44
Cmb(11)	2 - - 2 - -	18	0.0160	7.11E-01	43
Cmb(12)	- - - - 2 2	17	0.0158	7.70E-01	43
Cmb(13)	2 - 2 - - -	14	0.0163	9.45E-01	44
Cmb(14)	2 - - - - 2	13	0.0155	9.47E-01	42
Cmb(15)	- - 2 - 2 -	12	0.0167	9.86E-01	44

$(j = 1, ..., |C|)$, which are not independent of $D = d$ already obtained using Eq. (2), as shown in Table 4. Conversely, the $C(j) = v_{j_k} = rd(j,d)$ shown in Table 4 can be recognized as a set of partial rules of $RL = 1$.

After determining the partial rules of $RL = 1$ for $D = d$, partial rule candidates for $RL = 2$, $Cmb(k)$ can be constructed by the combination of partial rules of $RL = 1$ as follows: $Cmb(k) = (C(j_1) = rd(j_1, d)) \wedge (C(j_2) = rd(j_2, d)) \wedge Q$. Here, $Q = \bigwedge_{j=1, j\neq j_1, j_2, j_1 \neq j_2}^{|C|} (C(j) \neq rd(j,d))$ is necessary not to make a partial rule of more than $RL = 3$. Whether or not $Cmb(k)$ is a partial rule for $RL = 2$ can be determined via statistical testing, specifying the null hypothesis, H_0: $Cmb(k)$ is not a partial rule for $RL = 2$ (the alternative H_1: $Cmb(k)$ is a partial rule for $RL = 2$), and using $U(d)$. Table 7 shows the test results with the combination test table for $D = d = 2$ ($CTT(d = 2)$) in ascending order of the corresponding p-values. The number of patterns is ${}_{|C|}C_2 = 15$, whereas (d, j_1, j_2) denotes the pattern of $Cmb(k)$; $freq.(d, j_1, j_2)$ is the outcome frequency of the pattern among $|U(d = 2)|$, whereas the p-value of the freq. (d, j_1, j_2) is calculated using the probability $P(d, j_1, j_2)$ under H_0; and $freq.(p0)$ is a reference of the frequency corresponding to p-value$= p0$. In reality, if $freq.(d, j_1, j_2) > freq.(p0)$, the p-value of $Cmb(k) < p0$.

As Table 7 shows, there was a large discrepancy in p-value between $Cmb(9)$ and$Cmb(10)$, meaning H_0 of $Cmb(k)$ $(k = 1, ..., 9)$ should be rejected, that is, it should be determined to be a partial rule for $RL = 2$. Meanwhile, H_0 of $Cmb(k)$ $(k = 10, ..., 15)$ cannot be rejected, that is, it should not be regarded to be a partial rule for $RL = 2$. The test results in Table 7 fall in line with $R(2, 1)$ and $R(2, 2)$. Specifically, $Cmb(k)$ $(k = 10, ..., 15)$ is a straddled rule between two

Table 8. Example of adjacency matrix for Table 7.

-	$C(1)$	$C(2)$	$C(3)$	$C(4)$	$C(5)$	$C(6)$
$C(1)$	0	1	0	0	1	0
$C(2)$	1	0	1	1	1	1
$C(3)$	0	1	0	1	0	1
$C(4)$	0	1	1	0	0	1
$C(5)$	1	1	0	0	0	0
$C(6)$	0	1	1	1	0	0

rules. For example, $Cmb(10)$ can be made of like $u^C(3)$ or $u^C(4)$ in Table 3 by randomly determining $R(2,1)$ or $R(2,2)$.

Similarly, the rules of more than $RL = 2$ can be constructed by combining them with more rules of $RL = 1$. Nevertheless, it is difficult to statistically determine whether or not they are the partial rules since the probability of such partial rules is extremely low (see the probability of these with $RL = 4$ in section 3) and sufficient outcomes for testing cannot be obtained. Now, we can examine the relationships of the rules of $RL = 2$ in graph theory. Table 8 shows the adjacency matrix derived from Table 7. Here, the element $(1,2) = 1$ denotes the connection between $C(1)$ and $C(2)$ corresponding to $Cmb(4)$ in Table 7 and the element $(1,3) = 0$ the nonconnection between $C(1)$ and $C(3)$ corresponding to $Cmb(13)$. This matrix is symmetrical, with the diagonal set to 0.

The graph expression of Table 8 is shown in Fig. 2, where $C(1)$, for example, is denoted by $C1$ because of the software used [10]. The software could also induce the maximum cliques [11] in the graph as follows:

```
[[1]] + 3/6 vertices, named, from f88b50c: [1] C1 C2 C5
[[2]] + 4/6 vertices, named, from f88b50c: [1] C2 C3 C6 C4
```

The two induced cliques correspond to $R(2,1)$ and $R(2,2)$, respectively, since $R(2,1)$ with $RL = 3$ have three partial rules of $RL = 2$ ($_3C_2 = 3$), $R(2,2)$ with $RL = 4$ also having six ($_4C_2 = 6$), and both groups of rules with $RL = 2$ forming their own respective clique, which was confirmed by the results shown in Fig. 2.

We labeled the aforementioned if-then rule induction procedure as the Bayesian STRIM after inferring the cause CP from the observed results $U(d)$ and arranging the procedure, as shown in Fig. 3.

Let us apply the Bayesian STRIM to $U(D = 4)$ (Step1) generated by $R(4,1)$ and/or $R(4,2)$, which have both a duplicate point and a reduct point. In this case, $RT = \{rd(j, d = 4)|j = 1, ..., |C|, d = 4)\}$ has already been obtained (Table 4 (Step 2)). In Step3, $K = 6 - 1$, $_KC_2 = 10$ and $CTT(d)$ was created, as shown in Table 9, which indicated that H_0 of $Cmb(k)$ $(k = 1, ..., 6)$ should be rejected but that of $Cmb(k)$ $(k = 7, ..., 10)$ should not, given the large discrepancy in p-value between $Cmb(6)$ and $Cmb(7)$. Consequently, the adjacency matrix was created, as shown in Table 10 (Step 4). Meanwhile, the graph of the adjacency matrix was created (Fig. 4), and the cliques were obtained as follows:

```
[[1]] + 1/6 vertex, named, from f633af1: [1] C3
```

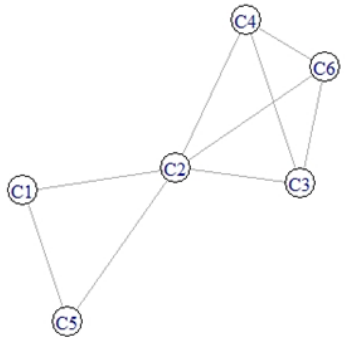


Fig. 2. Example of graph expression for the adjacency matrix presented in Table 8.

Step1	Derive $U(d) = \{u(i)\|u^{D=d}(i)\}$ $(d = 1, ..., M_D)$ from $U = \{u(i) = (u^C(i), u^D(i))\|i = 1, ..., N = \|U\|\}$.
Step2	Create $RT = \{rd(j,d)\|j = 1, ..., \|C\|, d = 1, ..., M_D)\}$, using $U(d)$.
Step3	Create $Cmb(k)$ $(k = 1, ..., {}_KC_2$, where K is the sum of the number of nonreducted $C(j)$ with $D = d$). Statistically evaluate whether $Cmb(k)$ is a partial rule of $RL = 2$, and create $CTT(d)$.
Step4	Create the adjacency matrix, $\{(C(j_1), C(j_2))\|j_1, j_2 = 1, ..., \|C\|\}$, using the statistical test result for $Cmb(k)$ in $CTT(d)$.
Step5	Create the graph of the adjacency matrix, induce the maximum cliques, and infer the if-then rule of CP for $D = d$.
Step6	Repeat Step1-5 from $d = 1$ to M_D.

Fig. 3. Example of procedures for the Bayesian STRIM.

```
[[2]] + 3/6 vertices, named, from f633af1: [1] C1 C2 C5
[[3]] + 3/6 vertices, named, from f633af1: [1] C2 C4 C6
```

Although [[1]] suggests that $C(3)$ is a rule of $RL = 1$, in reality, it is not, since $C(3)$ had already been reducted for $D = 4$ (Table 4). If this was not the case, $C(3)$ would be a rule of $RL = 1$, that is, the Bayesian STRIM can naturally induce a rule of $RL = 1$. Meanwhile, [[2]] and [[3]] suggest $R(4, 1)$ and $R(4, 2)$, respectively, meaning the Bayesian STRIM effectively induced the prespecified rule for $D = 4$ (Step5). The validity of the Bayesian STRIM for $D = d = 1, 3$ and 5 was also confirmed similarly as with $D = d = 2$ and 4 using the procedures shown in Fig. 3.

To examine the robustness of the Bayesian STRIM as a rule induction method, the following experiment was conducted:

Step 1) The data were resampled by N_B from the dataset of $N = 10,000$ generated in section 3 via a bootstrap method, and a new DT was formed.

Step 2) The Bayesian STRIM was applied to the new DT and whether or not it induced just enough prespecified rules for every $D = d$ was investigated.

Table 9. Example of $CTT(D = 4)$ for the PCH dataset presented in Table 5 ("-" denotes $C(j) \neq rd(j, d)$, $|U(d = 4)| = 1,013$).

Combination No. $Cmb(k)$	(d, j_1, j_2)	$freq.(d, j_1, j_2)$	$P(d, j1, j2)$ under H_0	p-value	$freq.(p0)$
$Cmb(1)$	4 - - - 4 -	86	0.0208	6.54E-28	43
$Cmb(2)$	- 4 - - 4 -	82	0.0196	5.42E-27	41
$Cmb(3)$	- 4 - - - 4	84	0.0217	1.92E-25	44
$Cmb(4)$	- - - 4 - 4	76	0.0208	1.01E-21	43
$Cmb(5)$	- 4 - 4 - -	73	0.0196	2.25E-21	41
$Cmb(6)$	4 4 - - - -	76	0.0217	1.17E-20	44
$Cmb(7)$	4 - - 4 - -	29	0.0208	3.63E-02	43
$Cmb(8)$	- - - 4 4 -	16	0.0188	7.11E-01	40
$Cmb(9)$	4 - - - - 4	17	0.0230	8.91E-01	46
$Cmb(10)$	- - - - 4 4	14	0.0208	9.32E-01	43

Table 10. Example of adjacency matrix for Table 9.

-	$C(1)$	$C(2)$	$C(3)$	$C(4)$	$C(5)$	$C(6)$
$C(1)$	0	1	0	0	1	0
$C(2)$	1	0	0	1	1	1
$C(3)$	0	0	0	0	0	0
$C(4)$	0	1	0	0	0	1
$C(5)$	1	1	0	0	0	0
$C(6)$	0	1	0	1	0	0

Step 3) Steps 1) and 2) were repeated $N_r = 100$ times.

Table 11 shows the results for step 3) arranged in terms of the rule induction rate (Ir). Here, p-value0 was uniformly used for a threshold value of rejecting H_0 of $Cmb(k)$ against all $D = d$ and creating the adjacency matrix as in Step 4, whereas with the examples shown in Tables 7 and 9, a large discrepancy in p-value was used. For example, in the row denoting N_B=5,000, p-value0= $1.0E-5$, and $Ir(D = 2) = 0.95$, whereas $Ir(D = 4) = 0.95$, which was in line with the conditions presented in Tables 7 and 9. The five induction failures at $D = 2$ were caused by the fact that the p-value of $Cmb(9)$ exceeded p-value0 and the H_0 of $Cmb(9)$ was not rejected. By contrast, the p-value of $Cmb(7)$ (Table 9) was less than p-value0 and the H_0 was rejected five times. However, as Table 11 shows, the Bayesian STRIM had the capacity to robustly induce prespecified rules by appropriately setting the number for N_B and using the individual p-value0 for every $D = d$.

5 Consideration of Differences between Bayesian STRIM and Other Methods Exploring for Causality

In this section, we examine the relationship between two Bayesian methods in terms of causal inference.

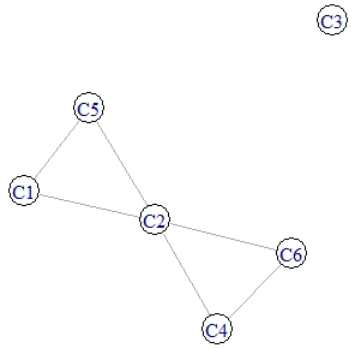


Fig. 4. Example of graph expression for adjacency matrix in Table 10.

Table 11. Examples of rule induction rates obtained via the bootstrap method.

N_B	p-value0	$Ir(D=1)$	$Ir(D=2)$	$Ir(D=3)$	$Ir(D=4)$	$Ir(D=5)$
4,000	1.0E-04	0.96	0.88	0.96	0.93	0.96
	1.0E-05	0.88	0.65	0.99	1.00	1.00
5,000	1.0E-04	0.96	0.97	0.99	0.87	0.98
	1.0E-05	0.98	0.95	1.00	0.95	1.00
	1.0E-06	0.95	0.69	1.00	1.00	1.00
6,000	1.0E-05	1.00	1.00	0.99	0.95	0.99
	1.0E-06	1.00	0.93	1.00	1.00	1.00
7,000	1.0E-05	1.00	1.00	1.00	0.95	1.00
	1.0E-06	1.00	1.00	1.00	1.00	1.00
8,000	1.0E-05	1.00	1.00	0.98	0.93	1.00
	1.0E-06	1.00	1.00	1.00	0.99	1.00

5.1 Relationship with MAP estimate

When $U(d)$ is given, the probability that the cause is the CP is given using Bayes' theorem as follows:

$$P(CP|D=d) = \frac{P(D=d|CP)P(CP)}{P(D=d)} = \frac{\frac{|U(d)\cap U(CP)|}{|U|}}{\frac{|U(d)|}{|U|}}$$
$$= \frac{|U(d)\cap U(CP)|}{|U(d)|} = Coverage(CP), \tag{5}$$

where $U(CP) = \{u(i)|u^C(i)\, satisfies\, the\, CP\}$. Equation (5) can be used for the MAP estimate [12] regarding the $P(CP)$ of its prior probability. If applied to the results shown in Table 5, $CndCP(2,1)$ with the maximum coverage will be chosen. However, $CndCP(2,1)$ is the partial rule with $RL=1$ of $R(2,1)$ or $R(2,2)$. The PCH generates the dataset generated from various types of partial rules, as shown in section 3. Meanwhile, the Bayesian STRIM estimates the original

rules after inducing the partial rules via a statistical test and constructing their connections according to their maximum cliques.

Besides Eq. (5), on applying Bayes theory to Eq. (1) on which the previous STRIM is based, the following is derived:

$$P(D=d|CP) = \frac{P(CP|D=d)P(D=d)}{P(CP)} = \frac{\frac{|U(d)\cap U(CP)|}{|U|}}{\frac{|U(CP)|}{|U|}}$$
$$= \frac{|U(d)\cap U(CP)|}{|U(CP)|} = Accuracy(CP), \quad (6)$$

which leads to selecting $CndCP(2,22)$, $CndCP(2,23)$, and so on (Table 5) as the rules. However, various problems related to selecting the rule candidates by referring to their coverage and/or accuracy have been reported in previous studies [5, 8].

5.2 Relationship with the BN [13]

Each attribute in a given DT is a random variable and its value or data are an outcome of the variable. The BN recognizes these random variables as vertexes of a graph, connects two of the vertexes with a directed acyclic edge to form an acyclic graph, and potentially presents a causal model. Once the dataset of the random variables is given, the BN identifies the causal model as a stochastic network using the dataset and can provide useful information regarding the variables by simulating the network under various conditions. The PC algorithm [13], which is regarded as one method for identification, first induces an undirected graph associated with the directed graph and then transforms the former into the latter using orientation rules (see [13] for details).

The input-output relationship of STRIM was presented in Fig. 1, whereas the relationship between random variables (e.g., in the case of $R(2,1)$ and $R(2,2)$) is shown in Fig. 5, where the rules were depicted in terms of a graph. Each edge of the condition part of the rule is given in a subset of the vertex, $\{C(1), C(2), C(5)\}$, $\{C(2), C(3), C(4), C(6)\}$, which are known as hyperedges, whereas the condition part is given in a hypergraph [14], $H = (C, F)$, where $C = \{C(j)|j = 1, ..., 6\}$ and $F = \{\{C(1), C(2), C(5)\}, \{C(2), C(3), C(4), C(6)\}\}$. On including the BN, most of the graphs connect two vertexes as an edge. However, the if-then rule requires the hypergraph to express itself in the graph expanding the edge into the hyper. Furthermore, only Fig. 5 does not express the PCH. Accordingly, it would appear to be difficult to use the BN to simulate the dataset generated by the if-then rule with PCH, except in the case of special rules with $RL = 1$,such as if $C(1) = d$ then $D = d$.

6 Conclusion

In this study, the previous if-then rule decision model, that is, the previous DG model applying the CCH was reviewed and improved to form the new DG model

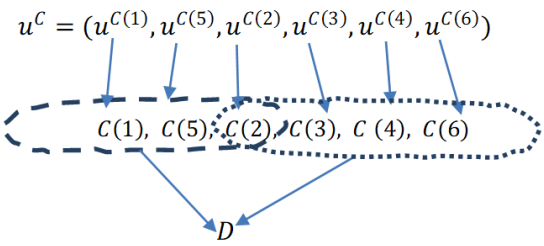


Fig. 5. Example of graph expression for $R(2,1)$ and $R(2,2)$.

applying the PCH. Specifically, the previous DG model involved the problem that it did not function appropriately in the case where the RL becomes longer because of the strict rules, whereas human beings tend to use such rules, making the second-best decision through referring to them. With this in mind, the new DG model incorporating the PCH was developed to ensure that it was closer to real-world datasets. The PCH dataset generated after prespecifying various types of if-then rules in the improved DG model was applied to the previous STRIM developed under the CCH dataset to examine its rule induction ability. Here, the principle of the rule induction performed by the previous STRIM is based on a statistical test, with specifying H_0: $P(D = d|CP) = P(D = d)$ and using U, and the CPs rejecting H_0 was adopted as the rule candidates. Based on the results, the following conclusions could be drawn:

1) The previous STRIM induced a large number of partial and/or straddle rule candidates with $RL = 1, 2, ...$ of the prespecified rules (see Table 5).
2) The method did not involve any appropriate strategy for arranging these rules in view of inducing original rules (see Table 6).

Then, this paper developed the previous STRIM into a new method applicable to PCH datasets and its validity of the new method was confirmed. Specifically, this paper reviewed the experimental results 1) and 2), studied the rule induction strategy on the basis of H_0: $P(CP|D = d) = P(CP)$ using $U(d)$, induced partial rules of CPs rejecting H_0 every $RL = 1$ and 2, assembled these CPs into the adjacency matrix and estimated the original rules using the maximum cliques in the graph derived from the matrix. The rule induction method was labeled the Bayesian STRIM after inferring the cause CP of the condition part from the effect $U(d)$ of the decision part of the if-then rule, and its procedure was arranged and summarized using illustrations of typical examples (Fig. 3). Furthermore, the robustness of the Bayesian STRIM was confirmed via a bootstrap method.

The relationships with MAP estimation and the BN were also studied in terms of the causal inference, specifically in relation to the if-then rule induction problem. Here, it was found that these methods are not generally applicable to the rule induction problem.

In future studies, we will focus on the following points:

(1) At present, the model assumes that $C(j)$ $(j = 1, ..., |C|)$ exhibits the same weight of importance $w(j) = 1.0$; however, both the actual $w(j)$ and the PCH must be incorporated in the model to ensure it is closer to real-world datasets.
(2) The Bayesian STRIM must be applied to real-world datasets in view of confirming its usefulness.

Acknowledgments

This work was supported by JSPS KAKENHI Grant Number JP20K11939.

References

1. Pawlak, Z: Rough sets; International Journal of Computer and Information Sciences (Springer), vol. 11, No. 5, pp. 341-356 (1982).
2. Zhang Q., Xie Q. and Wang, G.: A survey on rough set theory and applications; CAAI Transactions on Intelligence Technology, ELSEVIER, Vol. 1, pp. 323-333 (2016).
3. Matsubayashi, T., Kato, Y. and Saeki, T.: A New Rule Induction Method from a Decision Table Using a Statistical Test, In T. Li et al. (Eds.): RSKT 2012, LNAI 7414, pp. 81-90 (2012).
4. Kato, Y., Saeki, T. and Mizuno, S.: Considerations on Rule Induction Procedures by STRIM and Their Relationship to VPRS, In M. Kryszkieicz et al. (Eds.): RSEISP 2014, LNAI 8537, pp. 198-208 (2014).
5. Kato, Y., Saeki, T. and Mizuno, S.: Proposal of a Statistical Test Rule Induction Method by Use of the decision Table, Applied Soft Computing (ELSEVIER), Vol. 28, pp. 160-166 (2015).
6. Kato, Y., Saeki, T. and Mizuno, S.: Proposal of a Statistical Reduct Method for Decision Tables, In D. Ciucci et al. (Eds.): RSKT 2015, LNAI 9436, pp.1-13, Springer, Heidelberg (2015).
7. Fei, J., Saeki, T. and Kato, Y.: Proposal for a New Reduct Method for Decision Tables and Improved STRIM, DMBD 2017, LNCS 10387, pp. 366-378 (2017).
8. Kato, Y., Saeki, T. and Mizuno, S.: Considerations on the principle of rule induction by STRIM and its relationship to the conventional Rough Sets methods, Applied Soft Computing (ELSEVIER), Vol. 73, pp. 933-942 (2018).
9. Walpole, R. E., Myers, R. H., Myers, S. L. and Ye, K.: Probability and Statistics for Engineers and Scientists, Eighth edition, Pearson Prentice Hall, pp. 187-191 (2007).
10. `https://cran.r-project.org/web/packages/igraph/igraph.pdf`
11. Clique (graph theory) - Wikipedia.
12. Maximum a posteriori estimation - Wikipedia.
13. Spirtes, P., Glymour, C., Scheines, R. and Tillman R.: Automated Search for Causal Relations: Theory and Practice, In Heuristics, Probability and Causality: A Tribute to Judea Pearl, Dechter, R., Geffner, H. and Halpern, J. (Eds.), College Publications, pp. 467-506 (2010).
14. Hypergraph - Wikipedia.

Implementation of Neural Networks to Predict Crop Yield Production in Norwegian Agriculture *

Rashmi Gupta[1][0000−0003−4737−8775], Martin Engen[1][0000−0002−1121−0332], Erik Sandø[1][0000−0002−7191−5967], Benjamin Lucas Oscar Sjølander[1][0000−0001−6349−5408], Simon Arenberg[2], and Morten Goodwin[1][0000−0001−6331−702X]

[1] Centre for Artificial Intelligence Research (CAIR), Department of ICT, Faculty of Engineering and Science, University of Agder, Grimstad, 4879, Norway
rashmi.gupta@uia.no (R.G.), martin.engen@outlook.com (M.E.), esandoe@outlook.com (E.S.), benjamin@sjolander.no (B.L.O.S.), morten.goodwin@uia.no (M.G.)

[2] inFuture AS, Oslo, Norway simon.arenberg@infuture.no (S.A.)

Abstract. Crop yield prediction is a natural development of sustainable agriculture, which helps produce a rich amount of good quality food without depleting and polluting environmental resources. Crop yield production has many contributing environmental factors that can predict the wellness of crop production over a growing season. In this preliminary experiment, we identify the potential of machine learning algorithms in the agriculture domain with an increased amount of relevant data available. We mainly utilised four different sources of agriculture data, i.e., annual grants of the individual farmers, the total quantity of grain delivered by each farmer, the geographical location of the registered farms, and the weather information in the growing season. After preprocessing and filtering the utilised data corpus, we train a Deep Neural Network model (DNN) to predict the relative crop yield (i.e., in kg/10 000 m^2) for each farmer's farm location. We identify that the trained DNN model could predict crop yield with an average error of 930 kg/10 000 m^2, or 24% of the average crop yield per /10 000 m^2 for the utilised data corpus (i.e., 3828 kg). We implement the same model for further experiments in this research study. We compare the crop yields providing different grain types and substituting the weather information for one year with another to evaluate its impact on overall crop yield prediction. In conclusion, the reusable crop yield production dataset and the proposed novel model could meet the actual requirements for the prediction targets in this paper, providing further valuable insights for the research community.

Keywords: crop yield prediction · norwegian agriculture · deep learning · artificial neural network · deep neural networkis

* Supported by Norwegian Research Council-funded project 309876 KORNMO - production optimisation, quality management, and sustainability through the grain value chain.

Introduction

In Norway, a sustainable crop yield production highly depends on agro-climatic conditions, infrastructural developments in agriculture, annual grants for individual farmer, the persistence of rainfall, and the weather information [1]. It is becoming a significant challenge for farmers to produce an increased quantity and better quality of different grain types by increasing the overall global population [2]. In this paper, we focus on exploring deep learning models to predict crop yield production in Norway. It is our belief that the farmers could get the valuable insights to know the production improvements in particular grain type and quantity of crops in growing seasons based on geographical location and other environmental factors. In addition, it will improve food security and aid decision-making at various administrative levels.

In the past decade, we observed a steady decline in both the count of active grain farmers and the total landmass area used for grain farming [3]. In addition, we observed the increased work efficiency of the grain farmers at the same time, that leads to a total grain production at the national level. In 2019, the Norwegian farmers cooperators and Felleskjøpet (FK) agriculture organisation received a large investment into a new agriculture project. This project aims to increase the quantity of the crop yield and overall efficiency in Norwegian crop yield production system. In result, they combine the traditional farming methods with new digital technologies such as machine learning or deep learning for data harvesting in the growing season [4]. According to a survey on Norwegian farmers for the oats grain production, it has been investigated that the crop yields vary with several environmental factors such as weather, current and previous crop types, and type or quantity of fertiliser used for the particular grain [5] [6]. It is our belief that the early predictions for these environmental factors' interaction and effects may provide better understanding to the farmers to improve crop yield production. Although, the attempts to predict crop yield production with Artificial Neural Networks (ANN) have been made before [7], data for Norwegian agriculture has recently improved in terms of availability and granularity, e.g., grant applications [8]. With this in mind, we have gathered and pre-processed relevant publicly available agriculture data and trained an artificial neural network model to predict the crop yield of Norwegian grain farmers for this preliminary research study.

The major motivation of this preliminary study is to predict crop yield production across Norway. The primary purpose is to observe what data sources are publicly available to conduct this research study and how this data corpus can be utilised and evaluated using machine learning algorithms, such as neural networks. We carefully explored the collected data, such as weather information and geographical locations of harvested area of farms. Based on this motivation, we identify two major goals for this study i.e., (i) predicting the quantity of the crop yield production delivered by each farmer based on the geographical area allocated along with weather information, and (ii) build and train neural networks to predict the crop yield for farms in Norway and evaluate the overall performance of each farmer in the growing season.

1.1 State-of-the-Art to Crop Yield Prediction

Based on the goals identified earlier in this section, the authors investigated that the neural networks have emerged as the current state-of-the-art for crop yield prediction using weather and geographical data [9]. As can be observed in the study by Lieu et al. in 1999 [9], where the researchers used artificial neural networks for crop yield prediction and found that the ANNs can capture the nonlinear function for crop yields. They used a simple artificial neural network with three fully connected layers to predict corn yield production using soil data, weather data, and management data. The weather data included rainfall for each month during the season, the previous year, and the calculated GDD (Growing-Degree-Day) for the season. The management data included genetic data and fertiliser, planting density, and rotation factors in field. In total, the network was fed with limited i.e. 15 input variables and 20 units in the hidden layer. Consequently, ANN model predicted the yield of 60 validation samples with an RMS error of around 20% [1].

Crtomir et al. in 2012 [10], used ANN along with image analysis for Apple yield prediction by looking at the predicted number of fruits at four or five different stages of growth. By looking at maize production in east-central Indiana, USA, using data from 1901 to 1996, O'Neal et al. [11] utilised weather data and implemented an ANN and regression models to predict the maize yield with an RMS error of 10.5% [1]. The researchers found that the maize yield production is highly co-related with harvested season of the year and showed insignificant coefficients with precipitation and air temperature dataset. Consequently, they found neural networks outperformed the linear and quadratic regression models used on the precipitation and air temperature dataset. They found that the neural network performed better using a max-min coding scheme (i.e., linear scaling of input values between a maximum and minimum value) compared to unary, binary, or logarithmic coding schemes. We implemented the findings from [11] in our experiment and found less RMSE i.e. 24% of average crop yield per $/10\,000\,\mathrm{m}^2$, which we believe is better outcome for research community. In conclusion, this shows that deep learning can be effective for crop yield predictions, given that enough crop yield statistics are available.

We present our contributions thus: (i) an accurate crop yield prediction using geographical locations and weather information data; (ii) fusing and filtering multiple sources of data to further improve the crop yield prediction accuracy; (iii) predicting a particular grain type with better yield production among four different grain types; and (iv) identifying the effects of weather information on crop yield prediction. In this section, we presented the overview of this research study and motivated the idea behind this study. In addition, we define the current state-of-the-art approaches to the crop yield predictions using neural network models. In Section 2, we report on the history of Norwegian agriculture and

[1] It is difficult to determine exactly how different researchers have calculated RMS as a percentage or if they are using the same formula. The values presented here are the values the researchers have used themselves.

grain production. We also report the relevant studies which observed the environmental factors effecting crop yield production and agricultural improvements overall. Section 3 presents the methodology of this research study. Section 4 showcases the experiments to predict crop yield production and summarises the results. We then assess the obtained results against our earlier mentioned identified goals in this section. We conclude this research study and discuss further directions for future work in Section 5.

2 Related Work

2.1 Norwegian Agriculture

Norwegian agriculture has traditionally been family farming [14]. With the support of society and politicians, the goal is to reach national self-sufficiency based on the available natural resources. The government has designed the subsidy rates to compensate for any disadvantages to keep agriculture active and profitable. For example, land payments are differentiated by geography and type of agricultural production [15]. Although Norwegian agriculture has comprised smaller family farms spread out, there has been a decline in the number of farm holdings by 50% in the last three decades. Simultaneously, there has been an increase in the average size of farms, from 14.7 hectares in 1999 to 24.7 hectares in 2018 [16]. The most produced categories of food throughout the country include milk and milk products, meat, poultry, eggs, potatoes, and grains [14]. The readers can have the access to the distribution of the developed grain production and geographical area (in acres) used in Norway for agriculture from 2003 to 2019 discussed in [17].

2.2 Norwegian Grain Production

The Norwegian topography consists mainly of mountain masses, and as a result, only 3% of the total landmass is cultivated land (excluding Svalbard and Jan Mayen). Because of differences in climatic conditions, a minor part of this cultivated land can grow cereal for human consumption [16]. As per the research, Norway's eastern and southeastern part is best suited to produce food-grade grains. There are mainly four types of grain produced in Norway: wheat, barley, oat, and rye (and rye-wheat). We found that barley is the most grown grain in Norway because of the need for a shorter growing period [18]. Wheat requires a longer growing season compared to barley. For example, winter wheat resumes growing in the spring and is harvested in the summer. The result is the same type of grain, but winter wheat may give higher yields as it will resume growing earlier in the spring compared to spring wheat [19]. Oats thrive in cold and moist climates, which makes them well suited for cultivation in Norway. The vast majority, i.e., over 90% of oats grown, are used for animal feed, and 2% are used for human consumption [5]. Rye is the least grown grain in Norway, covering 2% of the total area used for grains [20]. Rye thrives in higher altitudes, but is mostly only grown in the east and southeastern parts of the country [20] [21].

2.3 Plant Growth Factors

Plant growth and wellness are affected by elements from its surroundings. According to Woodward F. Ian [22] and Oregon State University [23], the five main environmental factors affecting plant growth are light, temperature, water, nutrition, and crop rotation.

- **Light:** Light is a component of photosynthesis and is essential for overall plant growth. In Norwegian crops, the duration of light is particularly relevant; according to Åssveen and Abrahamsen, the duration of light in a day (day length) is more influential than temperature as growth factors [24].
- **Temperature:** Temperature affects growth in several ways. A rise in temperature triggers the germination process, so the temperature controls when the seedlings initially sprout [25]. The temperature also affects when crops such as winter wheat break dormancy to resume the growth in spring [26]. Crops will have different requirements for measuring the average degrees are needed before it is ripe for harvest [22] [23].
- **Water:** Together with light, water is a primary component of photosynthesis, and consequently, an essential factor for growth. For crops, water can come in the form of direct precipitation, humidity, or irrigation [22] [23].
- **Nutrition:** Plants need in total 17 essential chemical elements to grow. Three of the required components are found in air and water (carbon, hydrogen, and oxygen), while the soil must provide the rest [27]. Farmers can fertilise the soil, which adds materials containing nutrients to make these available to the plants. The roots absorb approximately 98% of the nutrients through soil water [28]. If the plant is under stress by extreme temperatures, drought, or low light, this can lower the plants' ability to absorb nutrients efficiently [29].
- **Crop rotation:** Crop rotation is the process of rotating alternative crops that are growing in a field from season to season. It builds an opportunity to create diversity on a field [30]. Bullock in 1992, shows an example of a 2-year crop rotation between maize and soybean, which resulted in 5% to 20% more maize yields [31]. Crop rotations also help in breaking disease cycles, and including plants in the legume family in a rotation. In result, it will help pulling the nitrogen from the air and lessen the need to fertilise nitrogen in the field [30].

Based on the environmental factors [22] [23], Engen et al. in 2021 [32] predicted a farm-scale crop yield production from multi-temporal agriculture data such as Sentinel-2 satellite images, weather data, farm data, grain delivery data, and cadastre-specific data. The researchers proposed a novel deep hybrid neural network to train this multi-temporal data. They combined the features of convolutional layers and recurrent neural networks to predict farm-scale crop yield production across Norway. In result, the proposed network could efficiently make the target predictions with the mean absolute error of 76 $\mathrm{kg}/10\,000\,\mathrm{m}^2$, which outperformed the baseline approach reported in the research study.

3 Research Methodology

With the understanding of Norwegian agriculture, grain production and plant growth factors explained in Section 2, we present the research methodology for this study (See Fig. 1). We explore the collection of multiple sources of data in Section 3.1. We present the technique to filter the utilised data in Section 3.2. In Section 3.3, we show the techniques to prepare the data such as normalisation and de-normalisation techniques before feeding the data for training the model. We align the state-of-the-art deep learning model [11] with the preliminary experimentation of this study in Section 3.4.

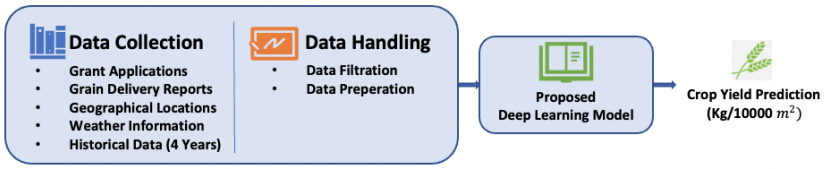


Fig. 1. Research methodology for this research study.

3.1 Data Collection

There is no readily available single dataset that can be downloaded and used for machine learning in crop yield prediction in Norwegian agriculture. Therefore, a large portion of this work has been collected from publicly available data sources that can be used for crop yield prediction.

– **Grant Applications and Grain Delivery Reports:** The primary data sources are the official public archives of farmer grant applications and grain deliveries from the Norwegian Agriculture Agency. As Norwegian grain farmers rely on subsidies, they must fill out yearly grant applications describing the land used for crop cultivation. These are used to build a base dataset that can integrate other data sources through each farmer's unique organisation number [2]. From the Norwegian Agriculture Agency, three different yearly reports serve as the base data for this research study. First, the grain delivery reports that include how much grain of different types, i.e., barley, oats, wheat, rye, and rye wheat, each farmer has sold in the last year. Second, the agriculture production subsidies from farmers regarding the area of cultivated land [3]. Third, we get a detailed report that links farmers' organi-

[2] where all farmers are registered in the official registers (*Brønnøysundregistrene*), giving them a unique organisation number

[3] these areas provide the relative crop yield for each farm and crop type from the year 2017 to 2019 for crop yield prediction

sation number to the used cadastral unit [4]. We served these data as a basis for precise geographic location [33].

- **Geographical Locations:** The address(es) for each organisation were gathered from the Brønnøysund Register Centre's Unit Register API. The addresses' global coordinates were obtained by querying a Geocoding API (accessible at https://www.geonorge.no/) from MapBox. These coordinates were subsequently used to query a different MapBox API to retrieve the elevation of each address.
- **Weather Information:** The weather is one of the main external factors that is crucial for farming [34]. As shown by [35], temperature is highly correlated to the eventual yield, and precipitation is relevant when there is no irrigation used [36]. The Norwegian Meteorological Institute (MET Norway) collects weather data across Norway and makes it publicly available through the Frost API (https://frost.met.no/). In this study, the weather at each farm was estimated using the readings from its nearest weather station [33]. As there are significantly more farms than weather stations, many farms share the same weather data. In addition, we found there were 353 weather stations with precipitation data and 288 with temperature data that could affect the overall prediction accuracy. We split temperature measurements into individual days, where a min, max, and arithmetic mean are stored. We downloaded the accumulated precipitation from each weather station with these measurements, resulting in one precipitation feature per day. In result, this helps to speed up training and reduce overfitting.

3.2 Data Filtration

In this section, we explore the collected data corpus for getting meaningful insights. The grain delivery dataset contains entries for each farmer per every year. Together with the grant application dataset, the grain deliveries allow us to look at each farmer's yield compared to the total area of a cultivated field (i.e., kg/10 000 m^2) is harvested. Through experimentation, we found the quantity of crop yield by different farmers closely depends on the farm's size (total area harvested). However, a few reports include negative yield, where crop yield is almost nothing even after harvesting maximum of farm-land area.

We see most farmers manage a yield somewhere between 0 and 10 tonnes per hectare (where 1 hectare = 10 000 m^2), or over 20 tonnes per 10 000 m^2 and higher. According to SSB [3], the average Norwegian farmer produced between 3 and 5 tonnes per 10 000 m^2 each season from 1997 to 2012, across the four major grain types. This average fits well with the majority of the samples in our dataset, and, shows that the observed samples with very high yield could be based on invalid or incomplete reports. Before using the data for training the neural network, we filter the data by removing outliers based on the distance from the mean yield. Because of the different grain types, the distributions are

[4] where a cadastral unit is an area of land, as specified in the official Norwegian cadastre (*matrikkelen*)

different. Thus, this filtering is done separately for each grain type. Removing data points over two standard deviations from the average yield results in a distribution that closely resembles a normal distribution (See Fig. 2).

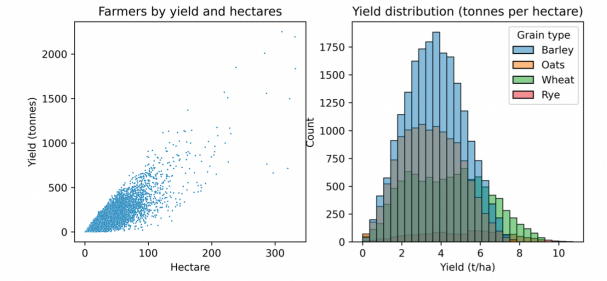


Fig. 2. Each farmers yield and yield distribution (per grain type), after filtering.

3.3 Data Preparation

All features are scaled or normalised before used as input to the neural network. Min-max normalisation applied to most features, meaning the values are linearly mapped to a value between 0 and 1, based on the maximum and minimum values of that feature. To keep temperature values consistent across all the weather features, we apply a pre-defined constant value for the minimum i.e., -30 degrees and maximum i.e., 30 degrees normalisation limits, meaning that values in the range [-30, 30] are linearly scaled to [0, 1] (See Equation 1). We use 0 and 10 mm as minimum and maximum values for precipitation, and for historical yield data, we use 0 and 10000.

$$normalised = (value - minimum)/(maximum - minimum) \tag{1}$$

The normalised values make the neural network's output difficult to interpret compared to real-world yield numbers. In result, we sometimes de-normalise the output values using the Equation 2.

$$value = output \times (maximum - minimum) + minimum \tag{2}$$

3.4 Proposed Neural Network Model in Predicting Crop Yields

Based on the earlier mentioned state-of-the-art technique by O'Neal et al. [11], ANNs outperform regression models for maize yield prediction. This motivated the authors to implement neural networks to predict crop yield production. In this research, the model was created using TensorFlow and is a simple network built up by dense layers. The network comprises one input layer, three hidden

dense layers with dropout in-between, and a final output layer. The input layer comprises all 762 features available, fed into the first hidden dense layer of 256 neurons. Next, a dropout layer with a rate of 0.1 to add random noise before another hidden dense layer of 64 neurons, another dropout layer with a rate of 0.25, leading to the last hidden layer of 64 neurons. At last, there is an output layer of one single neuron with linear activation, which outputs the predicted value. We used sigmoid, tanh, and relu activation functions in the hidden layers, where tanh proved to be the best for our experiment. In addition, we further improved the results using Adam optimiser with a learning rate of 0.0001 and a batch size of 512.

4 Experiments and Results

4.1 Predicting Total Crop Yield Production

In this initial experiment, we utilised weather data, historical yield, grants information, and farm data (the size and type of crop harvested) in earlier described neural network model. We split the utilised data in the ratio of 80:20 for training and validation set. As can be seen in Fig. 3, we obtained a validation loss of 0.008 mean absolute error, which roughly translates to an average error of 18 tonnes per each farmer. Both plots show the data ordered by yield, in increasing order. This shows the median prediction, and a minimum and maximum for bins of 50 samples each.

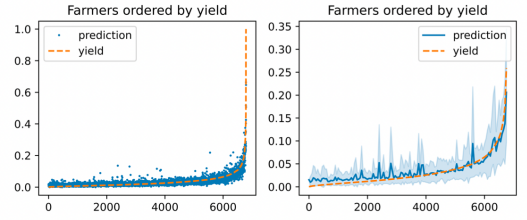


Fig. 3. Neural Network predictions compared with the actual crop yield on validation data.

4.2 Predicting Crop Yield Production based on Harvested Area

It is our belief the one can understand the quality of crop yield from the better measurement ratio between harvested area (i.e., 1 hec $= 10\,000\,\mathrm{m}^2$) and quantity of crop yield. Therefore, we set the target value to the yield per unit of harvested area in the neural network. As can be observed from Fig. 4, we got the predicted crop yields in increasing order per hectare as compared to the total crop yield

prediction. In result, we got a mean absolute error of 0.086, which further denormalised to 930 kg per10 000 m^2. This shows the 24% of the average yield per 10 000 m^2 in the dataset, i.e., 3828 kg. For the average farmer in our dataset, this equals 16.3 tonnes total yield, which is slightly better than predicting the total yield.

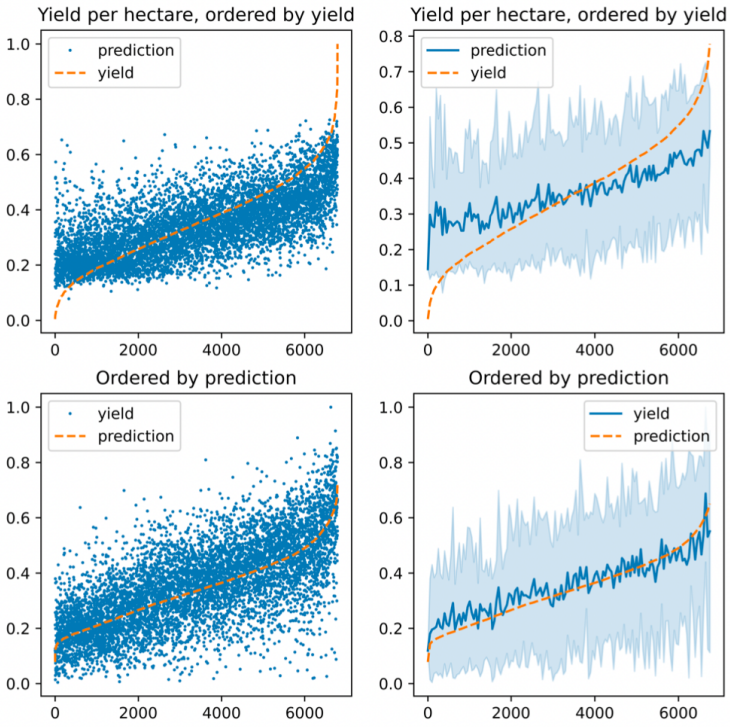


Fig. 4. Neural Network predictions for crop yield production based on harvested area.

4.3 Grants Used as a Proxy for Area

For this experiment, we utilised the Norwegian grain delivery data and grant reports data from the year 2013 to 2019. We observed the change in public grant reports data from 2017 on wards that include additional information, such as area harvested per grain type. From 2013 to 2016, the grant reports included financial details of a total harvested area and grants received per each farmer. In this experiment, we focus on considering the grants per total harvested area to compare with previously explained experiments. Where we faced a challenge to distinguish the quantity of each different grain type in that harvested area.

Because these grant reports also include grass land and the grain crops that are harvested for animal fodder use. Despite this, with the additional three years of grain delivery data, the model could train and predict reasonably well the farmer's relative production. It was challenging to directly compare the results described in Section 4.2, which used the area feature directly to the results using grants as area (See Fig. 5). Using grant reports provides additional insights, but it is less granular, and the amount of grants per hectare is fluctuating from year to year and from commune to commune. We see that the model reaches a loss of 0.115, which is 0.028 higher than what was achieved with the more detailed area in Section 4.2. We also saw a tendency of the model to guess the overall average, as shown in the bottom right plot of Fig. 5.

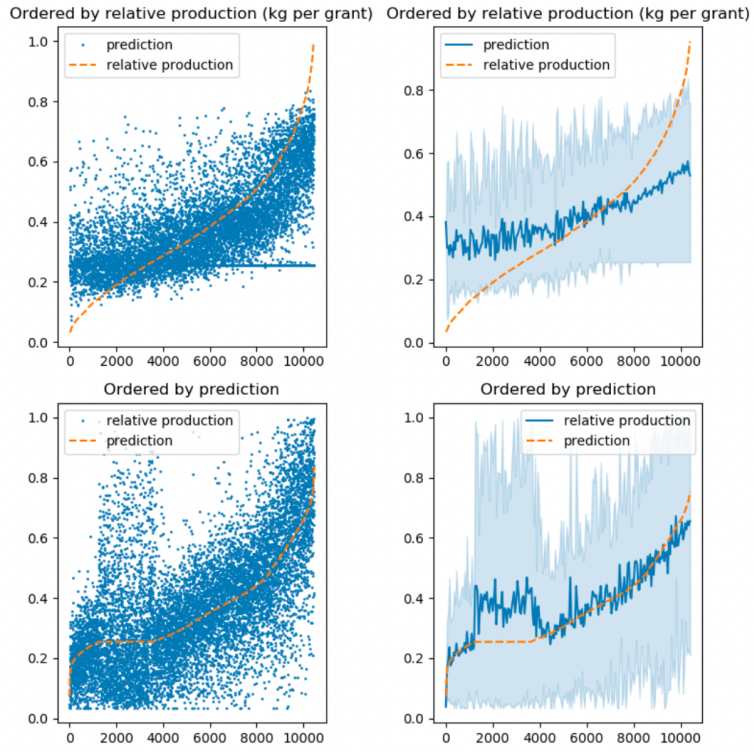


Fig. 5. Neural network using grants as a proxy for area to utilise all available data.

4.4 Predicted Crop Yield with Different Grain Types

To address the challenge, we investigated in Section 4.3; we focused on training the model to predict crop yield with different grain types in the harvested area

by utilising our collected data corpus. In result, we observed five different grain types, four alternative yields, and one for the actual yield per each farmer. This may identify the most suitable grain type for the farm-land that can further provide the knowledge-based assistance to the farmers, See Fig. 6. As can be observed from Fig. 6(a), the farmer chose to grow wheat, where the predictions are best for the rye and rye wheat, and wheat grain type. This shows that the farmer justifies the selection of the grain type in the growing season in the farm-land. On the other side, the other farmer (See Fig. 6(b)) selected to grow barley in the same growing season, which is clearly shows a worst prediction. Therefore, these predictions make the farmers understand the type of grain used to grow in a particular growing season to get the maximum of benefit in crop yield production.

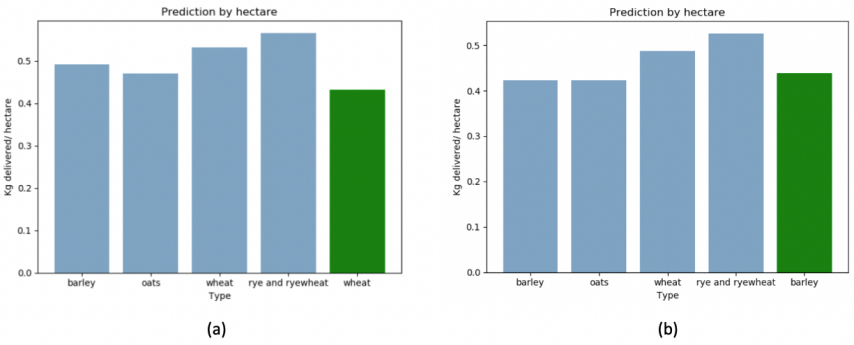


Fig. 6. Different crop yield predictions from two different farmers in a particular growing season.

4.5 Effects of Weather on Crop Yield Prediction

In this experiment, we utilised weather data along with one year of meagre crop yield data. We selected the crop yield dataset of the year 2018 as the base dataset, because we observed the least grain production in this year since 2013. In addition, we utilised weather data from the year 2015 [17][37][38]. Therefore, the selected data were fed to the model, and the results (See Fig. 7) show that the network can accurately predict the productions as compared to the production on average in 2018. We also see that the network predicted a 100 kg per hectare increase (34% increase) where we combined weather data from 2015, showing the consistency with that year's much higher grain production. However, the actual average yield from 2015 was even higher than the predicted 377 kg (denormalized) per hectare [38].

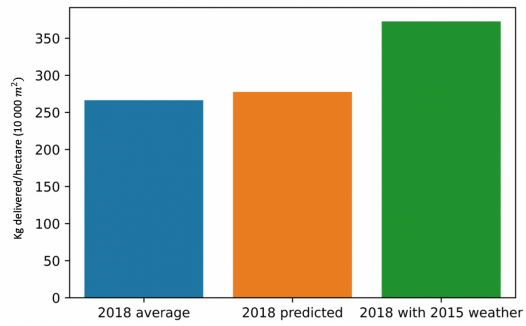


Fig. 7. The effects of applying weather data from a year with known good weather.

5 Conclusion and Future Work

This paper identified the potential challenges in predicting crop yield production. This research study found that the weather and climate factors lead to significant aspects determining crop yield. As a result, we utilised publicly available data from Norwegian grain farmers and geographic location data, daily weather temperatures, precipitation, and historical production numbers. Using a deep neural network with three hidden layers, we got a crop yield prediction accuracy of 930 kg per hectare. The neural network appeared to generalise well on weather data, even when tested on data from an earlier year that was not included during training. By changing the input manually, the network could be able to predict the quantity of the different yield types a farmer could have produced. This shows that it could be possible to use similar models to provide knowledge-based assistance to the farmers in deciding a particular grain type to plant in the growing season.

Based on this preliminary experimental study, we define our future work, which needs further exploration. Although data were gained and processed at several stages throughout, getting more data could improve learning. Thus, we list some potential data sources that can be considered for future work. First, the Sentinel-2 Satellites provide satellite images of the earth, usually at 5-10 days intervals, depending on the visibility conditions. This setup should give the model additional input to predict crop yields better. Second, we can use geographical location-based polygons for individual fields provided by the Norwegian Digifarm organization [5]. This data can be used to monitor the crops' condition during the growth period. Third, getting additional input from the farmers regarding the cultivation of their crops could provide better insight into how these factors affect yield. This could be data like watering, pesticide use, sow date, harvest date, and so on. Given a good learning model, it should improve performance, as these are all factors that should affect yield to a significant extent.

[5] Digifarm AS. From crop to the cloud. https://digifarm.io/.

References

1. Eltun, R.; Korsaeth, A.; Nordheim, O. A comparison of environmental, soil fertility, yield, and economical effects in six cropping systems based on an 8-year experiment in Norway. *Agric. Ecosyst. Environ.* **2002**, *90*, 155–168
2. Klaus G.G. Food quality and safety: Consumer perception and demand. *Eur. Rev. Agric. Econ.* **2005**, *32*, 369–391, https://doi.org/10.1093/eurrag/jbi011
3. SSB. Korn og oljevekster, areal og avlinger, https://www.ssb.no/jord-skog-jakt-og-fiskeri/statistikker/korn/aar-forelopige/2020-01-28. Last accessed 23 Nov 2020.
4. Aasvang, A. S. Nationen - Felleskjøpet får støtte til kornsatsing, https://www.nationen.no/landbruk/felleskjopet-far-stotte-til-kornsatsing/. Last accessed 24 Nov 2020.
5. Havre, A. Norwegian Digital Learning, https://ndla.no/subject:41/topic:1:188696/topic:1:189185/resource:1:82136. Last accessed 1 Dec 2020.
6. Brøndbo K. Havre-NM 2019 - FoU - Alt om norsk havre, https://ndla.no/subject:41/topic:1:188696/topic:1:189185/resource:1:82136. Last accessed 1 Dec 2020.
7. Klompenburg, T.V.; Kassahun, A.; Catal, C. Crop yield prediction using machine learning: A systematic literature review. *Comput. Electron. Agric.* **2020**, *177*, 105709.
8. Norwegian Agriculture Agency. Produksjons- og avløsertilskudd til jordbruksforetak søknadsomgang 2019, https://data.norge.no/ datasets/e341762d-311b-46f1-be25-078570f63204.. Last accessed 1 Dec 2020.
9. Goering, C.E.; Liu, J. (2000). Neural networks for setting target corn yields. Transactions of the ASABE, 44, 705-713.
10. Črtomir, R.; Urška, C.; Stanislav, T.; Denis, S.; Karmen, P.; Pavlovič, M.; Marjan, V. (2012). Application of Neural Networks and Image Visualization for Early Forecast of Apple Yield. Erwerbs-Obstbau, 54, 69-76.
11. O'Neal, M.R.; Engel, B.A.; Ess, D.R.; Frankenberger, J.R. (2002). AE—Automation and Emerging Technologies: Neural Network Prediction of Maize Yield using Alternative Data Coding Algorithms. Biosystems Engineering, 83, 31-45.
12. Khaki, S.; Wang, L.; Archontoulis, S.V. A CNN–RNN Framework for Crop Yield Prediction. *Front. Plant Sci.* **2019**, *10*, 1750.
13. Khaki, S.; Pham, H.; Wang, L. Simultaneous corn and soybean yield prediction from remote sensing data using deep transfer learning. *Sci. Rep.* **2021**, *11*, 11132.
14. Tine. Family Farming: The Key to Food Production in Norway, https://www.tine.no/english/about-tine/family-farming-the-key-to-food-production-in-norway. Last accessed 23 Nov 2020.
15. Bondelaget. Norwegian Agriculture—The Norwegian Model for Agriculture and Agricultural Policy. https://www.bondelaget.no/getfile.php/13894650-1550654949/MMA/BilderNB/Illustrasjoner/NorwegianAgricultureEN.pdf. Last accessed 17 Dec 2021.
16. Kildahl, K. Nine Facts about Norwegian Agriculture—Nibio, https://www.nibio.no/en/news/nine-facts-about-norwegian-agriculture. Last accessed 17 Dec 2021.
17. Brød og Korn. Kornproduksjon i Norge, https://brodogkorn.no/fakta/kornproduksjon-i-norge/. Last accessed 18 Dec 2021.
18. Norwegian Digital Learning Arena. Bygg, https://ndla.no/subjects/subject:41/topic:1:188696/topic:1:189185/.
Last accessed 18 Dec 2021.

19. Holtekjølen, A.K.; Uhlen, A.; Holtet, E.K.; Hvete, E.A. Store Norske Leksikon, http://snl.no/hvete. Last accessed 18 Dec 2021.
20. Rug, A. Norwegian Digital Learning, https://ndla.no/nb/subject:41/topic:1:188696 /topic:1:189185/. Last accessed 18 Dec 2021.
21. Encyclopaedia Britannica, Rye, https://www.britannica.com/plant/rye. Last accessed 18 Dec 2021.
22. Woodward, F.I. *Climate and Plant Distribution*; Cambridge University Press: Cambridge, UK, 1987; pp. 1–174
23. Oregon State University. Environmental Factors Affecting Plant Growth, https://extension.oregonstate.edu/gardening/techniques/environmental-factors-affecting-plant-growth. Last accessed 19 December 2021.
24. Åssveen, M.; Abrahamsen, U., Varmesum for sorter og arter av korn. *Grønn Forsk.* **1999**, *2*, 55–59.
25. Wahid, A.; Gelani, S.; Ashraf, M.A.; Foolad, M.R. Heat tolerance in plants: An overview. *Environ. Exp. Bot.* **2007**, *61*, 199–223.
26. Soureshjani, H.K.; Dehkordi, A.G.; Bahador, M. Temperature effect on yield of winter and spring irrigated crops. *Agric. For. Meteorol.* **2019**, *279*, 107664.
27. Burns, R.G.; DeForest, J.L.; Marxsen, J.; Sinsabaugh, R.L.; Stromberger, M.E.; Wallenstein, M.D.; Weintraub, M.N.; Zoppini, A. Soil enzymes in a changing environment: Current knowledge and future directions. *Soil Biol. Biochem.* **2013**, *58*, 216–234.
28. Doerr, S.H.; Shakesby, R.A.; Walsh, R.P. Soil water repellency: Its causes, characteristics and hydro-geomorphological significance. *Earth-Sci. Rev.* **2000**, *51*, 33–65.
29. Farooq, M.; Wahid, A.; Kobayashi, N.; Fujita, D.; Basra, S.M. Plant drought stress: Effects, mechanisms and management. *Agron. Sustain. Dev.* **2011**, *29*, 185–212.
30. Grainsa and Rhino Reloaded. Why crop rotation is important in wheat production, https://www.grainsa.co.za/ why-crop-rotation-is-important-in-wheat-production/. Last accessed 20 December 2021.
31. Bullock, D.G. (1992) Crop rotation. Critical Reviews in Plant Sciences, 11, 309-326.
32. Engen, M.; Sandø, E.; Sjølander, B.L.O.; Arenberg, S.; Gupta, R.; Goodwin, M. Farm-Scale Crop Yield Prediction from Multi-Temporal Data Using Deep Hybrid Neural Networks. Agronomy 2021, 11, 2576. https://doi.org/10.3390/agronomy11122576
33. Sjølander, B.; Sandø, E.; Martin, M. Neural Network for Grain Yield Predictions in Norwegian Agriculture. 2020, https://uia.brage.unit.no/uia-xmlui/handle/11250/2823807. Last accessed 20 December 2021.
34. Ngoune, L.T.; Shelton, C.M. Factors affecting yield of crops. In *Agronomy-Climate Change & Food Security*; IntechOpen: Rijeka, Croatia, 2020; doi:10.5772/intechopen.90672.
35. Johnson, D.M. An assessment of pre- and within-season remotely sensed variables for forecasting corn and soybean yields in the United States. *Remote Sens. Environ.* **2014**, *141*, 116–128.
36. Kukal, M.S.; Irmak, S. Climate-Driven Crop Yield and Yield Variability and Climate Change Impacts on the U.S. Great Plains Agricultural Production. *Sci. Rep.* **2018**, *8*, 3450.
37. SSB, Jordbruk, https://www.ssb.no/ jord-skog-jakt-og-fiskeri/faktaside/jordbruk. Last accessed 21 Dec 2021.
38. SSB. Tabell - Totalavling og avling i kilo per dekar av ulike kornslag. Fylke, https://www.ssb.no/150124/ totalavling-og-avling-i-kilo-per-dekar-av-ulike-kornslag. fylke. Last accessed 21 Dec 2021.

Predicting Student Performance of Online Courses with Deep Learning and NLP from Texts in Portuguese (pt-br)

Armindo Antonio Guerra Junior and Luciana de Oliveira Rech

Department of Informatics and Statistics (INE)
Federal University of Santa Catarina (UFSC)
Florianópolis - SC, Brazil
armindogjr@gmail.com, luciana.rech@ufsc.br

Abstract. According to the Analytical Report on Distance Learning in Brazil (EAD), the number of remote learning courses in various modalities, especially in tertiary education, has grown significantly over the past years. Alongside this trend, dropout rates of students are reaching 75% in technical courses and around 40% in tertiary education. One of the factors causing the increasing dropout rates is the low academic performance of students culminating in their respective failures [2]. This work proposes to use Natural Language Processing to help solve some of these problems. We propose using (NLP) and Deep Learning as methodologies for supervised classification of texts produced by distance learning students to predict low academic performance and allow educational institutions to react appropriately.

Keywords: Deep Learning · Natural Language Processing · Student Performance

1 Introduction

According to the analytical report on remote learning in Brazil, published in 2018 by the Brazilian Association of Distance Education (Abed), the number of undergraduate and specialization courses in the distance modality (EAD) is growing continuously. Alongside this, according to the same report [1], dropout rates of students are increasing, reaching 75% in technical courses and around 40% in tertiary education. The discussion surrounding the main factors that cause these failure rates is divided. Some authors point out social, institutional, and even personal factors [1]. However, a factor that stands out significantly, as a cause of these dropouts, is the student's low academic performance [2]. These high dropout rates in the context of courses offered by the government result in the loss and waste of public resources. Therefore, independent of the reasons that lead to these high dropout rates, it is essential that the educational society, as well as the involved managers, think of strategies to combat this problem.

One possibility to analyze the failure and dropout of students is through the data that the students themselves record on the online learning platforms. A specific and abundant type of data on these respective platforms is texts, produced

in discussion forums, homework, essays, or even in social interaction between students. Natural Language Processing, or simply NLP is a computational tool to analyze texts. NLP is a multidisciplinary area that encompasses Mathematics, Computer Science, Artificial Intelligence, and Linguistics to analyze problems related to the generation, interpretation, and manipulation of human language.

Due to the massive amount of text that is produced on the daily basis in blogs, news portals, social networks, etc., NLP currently receives a lot of attention from researchers and has become a heavily data-driven field. Over time, various Machine Learning approaches, that take advantage of the large quantity of data to map patterns, have been widely used in NLP applications. Among them, we can highlight Naive Bayes, k-Nearest Neighbor (KNN), Hidden Markov Models, Conditional Random Fields, Decision Trees, Random Forest, and Support Vector Machines (SVMs). However, for efficiency and assertiveness reasons, these approaches have recently been partially replaced, or at the very least improved, by Deep Neural Networks [4].

Deep Neural Networks, also known as Deep Learning, have achieved remarkable results in several fields of human knowledge including, but not restricted to, Computer Vision, Speech Recognition, and Text Classification [3]. For Text Classification, which considers texts as predictor variables in a supervised Machine Learning model, two types are applied to lead to good results; Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) [3]. CNNs can learn a local response from temporal or spatial data, but they cannot learn sequential correlations. In contrast, RNNs are specialized in sequential modeling, but cannot extract features in parallel. According to [3], both models, when used together and with the addition of other techniques, can present significant results in text classification.

Therefore, the present work has the main objective to apply Text Classification techniques, using network architectures such as CNNs, RNNs, among others, to predict the possible pass/fail of students to identify, in some cases prematurely, one of the main causes of the high dropout rates.

2 Related Work

The area of research related to the prediction of student performance is multidimensional and can be explored and analyzed through various perspectives, such as applying statistical techniques to predict dropouts directly, or analyzing and predicting the factors that most influence dropouts, such as the failures [19].

In the past, several works used Supervised Learning algorithms to classify students who dropped out of school, a good approach to the topic was discussed in [7] and [6]. In both works, the techniques that stood out the most were Naive Bayes, Association Rules Mining, RNA-Based Algorithms, Logistic Regression, CART, C4.5, J48, Bayes Net, Simple Logistic, JRip, Random Forest, and ICRM2. However, when the addressed problem is specifically classification, two methods were widely used with satisfactory results, Neural Networks and Decision Trees [8]. The advantage of Neural Networks, for example, is that they

can detect complex nonlinear relationships between dependent and independent variables [9], while Decision Trees stand out due to their simplicity and interpretability for mapping the searched patterns [10].

Specifically, on classroom performance, the authors of [8] carried out a systematic review of the literature to observe the characteristics that substantially influence the phenomenon. We can highlight studies that implemented machine learning techniques to analyze student behavior and predict the risk of failure [20] and [21], for example. The authors [22] used machine learning techniques sequentially in the timeline to predict students at risk of failing in the second, fourth and ninth weeks of the first year of engineering. Implementing a logistic regression model, they achieved an accuracy of 98% in the ninth week. In addition, the same authors performed several other tests, including the application of Support Vector Machines (SVM), Naive Bayes Classifier, Decision Tree, K-Nearest Neighbor, and Multi-Layer Perceptron, to identify the best prediction method. Another work that successfully applied logistic regression was [23]. Some works pointed out that student engagement is also a useful and important parameter to map student behavior, stating that more engaged students perform better [24], [25], an intuitive fact that was proven by the authors.

This present work differs from the others mentioned above as it purely applies text classification to the problem of predicting student failure in online courses, using Deep Learning and recent Natural Language Processing techniques.

3 Data Preparation

As a source of data for this work, database tables from students' distance learning courses at the Federal University of Santa Catarina (UFSC) were used. We sought to work with a final base, where students cannot be directly identified. Even with such precautions, a project was submitted to the Ethics Committee for Research with Human Beings at UFSC (CEPSH), to formalize the research and reinforce responsibility related to data treatment. After processing and evaluating the project by CEPSH, we received authorization to access and process the data.

3.1 Database

The complete database consists of 419 tables, of which 7 were used. After processing, the final database consists of 50,356 records, representing 259 courses and 3,492 students. Each record corresponds to a text written by the student in the forums of the platform and was linked with the respective outcome of the student in the course to which the forum is linked. A student was considered **Passed** when the grade achieved was equal to or greater than 6, in which case the student receives the label 1. Students with a grade lower than 6 were considered as **Failed** and received label 0. A diagram of the tables used is shown in Figure 1.

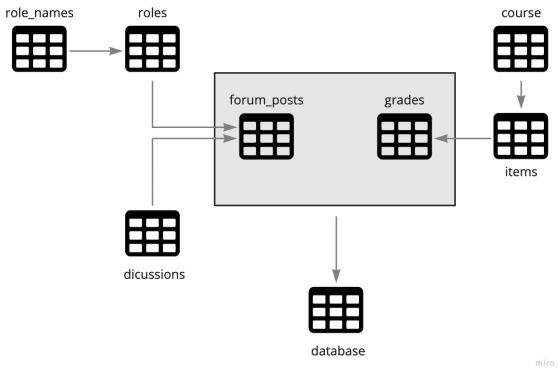


Fig. 1. Diagram of the tables of the database used

3.2 Cleaning, Treatment and Organization of Data

To prepare the student texts, in order to use them with Deep Learning algorithms, some data cleaning, and treatment techniques were applied. Punctuations, special characters, HTML tags, e-mails, URLs, phone numbers were removed. Portuguese words that when removed from the sentence do not affect general understanding (stopwords) were also excluded. Subsequently, each word of each text was separated (tokenization), to transform them into unique numerical values (IDs). And finally, empty lines were also removed. Figure 2 shows an example text of one of the students without treatment, clean and tokenized and in numeric format, respectively.

```
1 base_curso.msg[1]
```

```
'Toda essa desconfiança com um diploma EaD gerou-me uma dúvida: o que diz o diploma? Estará nele descrito claramente c
urso a distância? Não que eu me importe, mas o mercado sim! Será que a mudança, o caminho para a aceitação deste tipo
de ensino não deveria partir daí? Sem distinção, sem preconceitos! '
```

```
1 sentences[1]
```

```
'toda desconfiança diploma ead geroume dúvida diz diploma estará nele descrito claramente curso distância importe merc
ado sim mudança caminho aceitação deste tipo ensino deveria partir daí distinção preconceitos'
```

```
1 print(sequences[1])
```

```
[231, 5015, 3881, 185, 21781, 85, 221, 3881, 1503, 3609, 2280, 3224, 36, 214, 17746, 80, 37, 437, 520, 2281, 357, 184,
70, 302, 238, 2030, 5619, 6126]
```

Fig. 2. Text throughout treatment

The texts submitted by the students do not follow the requirements in regards to the word count, as such some categories were created to analyze the distribution of the word count in the respective categories. Figure 3 represents the distributions.

Most students texts are distributed among the categories **'between 10 and 50 words'**, **'between 50 and 100 words'** and **'between 100 and 200**

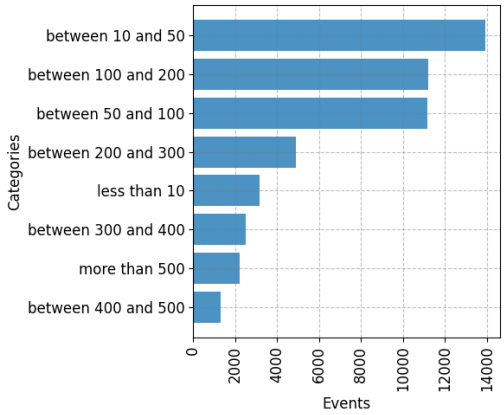


Fig. 3. Distribution of word count by categories

words'. Due to the amount of data available, we chose to apply the Deep Learning algorithms to these thee categories. We observed that the number of students who failed the subject decreased with increasing word count. For this reason, we apply the algorithms in each category separately to avoid bias at the time of learning. The exact distribution of these 3 categories can be seen in Figure 4.

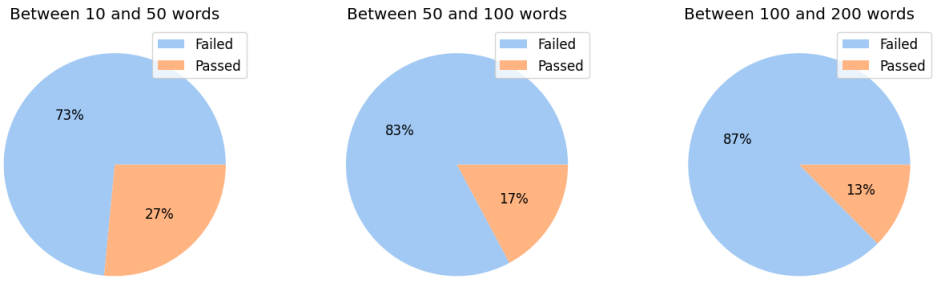


Fig. 4. Students passed x failed in main categories

Words that appear rarely, such as names, for example, receive a mark "out of vocabulary" (OOV) to be disregarded when training the Embedding Matrix. Texts with fewer words were completed with zeros until they became the same size as the largest sentence (padding) to guarantee that in the final process all sentences are of the same size.

3.3 Word Embeddings

To use computational methods for text processing and analysis, it is necessary to convert them into a numerical format suitable for the respective technique that will be used. In the first works surrounding NLP the vector space model was normally used, in which the values of individual words were estimated using the weighting method (based on the frequency with which the words appear) forming the vector of characteristics of each document which is known as the Bag of Words (BOW) Technique [12]. However, given the current amount of data, the size of the used vocabularies, and the fact that BOW is not efficient at capturing semantic information, the researchers investigated other ways of representing documents and words in a dense, low-dimensional, semantically significant vector space [12]. One method that has recently attracted attention is the distribution hypothesis, which assumes that contextually similar words have similar semantics [11]. Word representation methods based on the distribution hypothesis are mainly divided into three types [11]:

1. Matrix-based representation, also known as distributive representation;
2. Cluster-based distributed representation;
3. Distributed representation based on neural networks, also known as Word Embeddings.

Among the Word Embeddings techniques, one that has stood out for obtaining significant results is *Word2vec* [13], which is a distributed representation learning algorithm to learn continuous dense vector representations for words in low-dimensional vector space. The Algorithm consists of the joint use of two related models: the continuous bag of words model (CBOW) and the Skip-Gram model. CBOW predicts the current word from its surrounding context words in a sentence within a window centered on the current word, while the Skip-Gram model predicts the surrounding context words of a given word in a sentence within a symmetrical window. An important feature of this type of approach is that vectors of words in vector space are close to each other when the corresponding words are semantically similar to each other. This gives us the benefit of being able to infer semantically similar words by comparing the distance between the word vectors created by *Word2vec*. Then using *Word2vec* it is possible to create an Embedding Matrix that maps each word index to its corresponding embedding vector.

It is common to use Embedding Matrix pre-trained by other research groups. Even this training has been the subject of entire dedicated work, as is the case of NILC-Embeddings [14], which is a repository for the storage and sharing of vectors of words generated for the Portuguese language, whose objective is to promote and make ready-made vector resources accessible. In the present work, we chose to train our Embbeding Matrix, as we believe that the corpus extracted from the forums of the distance learning courses has intrinsic characteristics that would be lost when adopting another approach. So we did it, using the Python implementation of the **Gensim** tool version 3.8.3, which is an open-source library for modeling unsupervised topics and NLP. The **Fig. 5** provides an example of

the application of the Embedding Matrix to obtain words similar to the word "aluno" in the corpus used in this work.

```
In [47]: 1 word_vectors.similar_by_word("aluno")

Out[47]: [('estudante', 0.8087526559829712),
          ('alunos', 0.7965363264083862),
          ('educador', 0.7912662029266357),
          ('educando', 0.761611819267273),
          ('educandos', 0.7417762875556946),
          ('professor', 0.715225875377655),
          ('plateia', 0.6950972080230713),
          ('ensinoaprendizagem', 0.6872051358222961),
          ('autodidata', 0.6790434122085571),
          ('discente', 0.6764541268348694)]
```

Fig. 5. Similarity by words using *Word2vec*

After we trained the Embedding Matrix, we can use it for the initial weights in the first layer of the neural network.

4 Predicting Student Performance

Text classification can be considered a sequential modeling task, for this reason, Recurrent Neural Networks (RNNs) [17] are often used. However, for long data sequences, traditional RNNs suffer from two problems inherent to their functioning, explosion and gradient dissipation. In this context, the Long Short Term Memory (LSTM) [15] emerged, which are RNNs endowed with long-term memory units, which satisfactorily solve the problems with the gradient. Bidirectional long-term memory (BiLSTM) [16] is a natural evolution of LSTM networks because they are able to traverse the sequences in both temporal directions. For this reason, BiLSTM is generally superior to LSTM for working with texts.

The vector representation of texts usually has a high dimension. The high-dimensional vector as the input to an LSTM will cause a sharp increase in the parameters of the network and make it difficult to optimize it. The architecture of Convolutional Neural Networks (CNNs) [18] can be useful to extract features from data while reducing dimensionality. For that reason, the convolution operation can be used in this context. Therefore, the LSTM, CNNs, and BiLSTM architectures will be tested to classify the texts used in this work. The following sections will present the test results. The pseudo-code **Algorithm 1** summarizes the order in which the experiments will be conducted.

All neural network architectures were implemented in Python version 3.7.10, using API Keras version 2.4.3 over Framework Tensorflow in version 2.4.1.

4.1 Evaluating Metrics

As the number of failed students is significantly lower than the students that passed, we apply SMOTE for load balancing. With balanced data, Accuracy

Algorithm 1: Pseudo-code for text classification

```
1: Data load
2: Cleaning, organization and labeling
3: Training and creation of the Embedding Matrix
4: Separation into categories by word number distribution
5: load balancing
6: Application of algorithms (MLP, LSTM, CNN, CLSTM, C-BiLSTM)
7: Evaluation of results
```

was used as an evaluation metric to measure the overall performance of the classification. Accuracy is calculated as the ratio between the number of correct predictions to the total number of predictions:

$$Accuracy = \frac{TruePositive + TrueNegative}{TotalSample} \tag{1}$$

Another metric that can be used is Area Under The Curve (AUC) which is calculated about the Receiver Operating Characteristics curve (ROC). ROC is a curve that measures performance for classification problems at various threshold settings and is a probability curve, while AUC represents the degree or measure of separability. It shows how much the model is capable of distinguishing between classes. The ROC curve is plotted with the True Positive Rate (TPR) against the False Positive Rate (FPR) where TPR is on the y-axis and FPR is on the x-axis.

Precision is the fraction of relevant instances among the retrieved instances, while Recall is the fraction of relevant instances that were retrieved. For a classifier dedicated to binary classification, f1_score is an important indicator that combines Recall and Precision as follows:

$$f1\_score = \frac{2 * Precision * Recall}{Precision + Recall} \tag{2}$$

4.2 Parameter Settings

Data was separated into 80% for training, 10% for validation, and 10% for testing. During training in the Embedding layer, the input string is defined as the nth word in the vocabulary. The Embedding Matrix was trained from the available corpus with a dimension of 100. The memory dimension of the LSTM and BiLSTM networks was set to 128 with 50 filters of size 10 in the convolutional layers. The training batch size for all datasets was 256. The dropout rate in most tests was 0.6. A backpropagation algorithm with the Adam stochastic optimization method was used to train the network over time with a learning rate between 0.001 and 0.0001. The loss function used was Binary Cross Entropy. After each training period (epoch), the network is tested with the validation data.

5 Comparison and Analysis of Results

Below are the comparison tables of the application of Deep Learning network architectures to classify texts by EAD students at UFSC. The following architectures were used; Multilayer Perceptron (MLP), Convolutional Neural Networks (CNN), Long Short Term Memory (LSTM), and Bidirectional LSTM. CNN and LSTM (C-LSTM) and CNN and BiLSTM (C-BiLSTM) were used in a hybrid way.

In all scenarios, shown in **Table 1**, **Table 2** and **Table 3**, using accuracy as the main evaluation metric, C-BiLSTM showed better performance. This is justified by the fact that in this type of architecture the convolutional networks serve the purpose of extracting the main characteristics of the analyzed vectors while acting in the reduction of the dimension of the same vectors and Bi-LSTM work by traversing the vectors in both temporal directions, accentuating the semantic characteristics already calculated by the Embeddings Matrix. Both, when used together, naturally present good results. The **Fig. 6**, **Fig. 7** and **Fig .8** present the graphs with a history of accuracy and error over the epochs for the 3 categories tested by applying the C-BiLSTM architecture, which obtained the best results.

Table 1. Results of experiments with texts between 10 and 50 words

-	Accuracy	Loss	AUC	f1_score
MLP	0.81	0.42	0.81	0.80
CNN	0.84	0.44	0.84	0.83
C-LSTM	0.84	0.45	0.84	0.82
C-BiLSTM	0.86	0.40	0.86	0.85

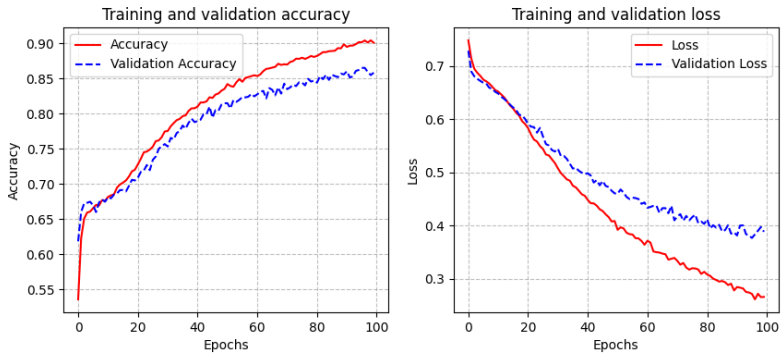


Fig. 6. History of the accuracy and error of the C-BiLSTM architecture applied to the training and validation data for the 10 to 50-word category

Table 2. Results of experiments with texts between 50 and 100 words

-	Accuracy	Loss	AUC	f1_score
MLP	0.94	0.17	0.95	0.94
CNN	0.94	0.22	0.94	0.94
C-LSTM	0.93	0.23	0.93	0.92
C-BiLSTM	0.95	0.21	0.94	0.93

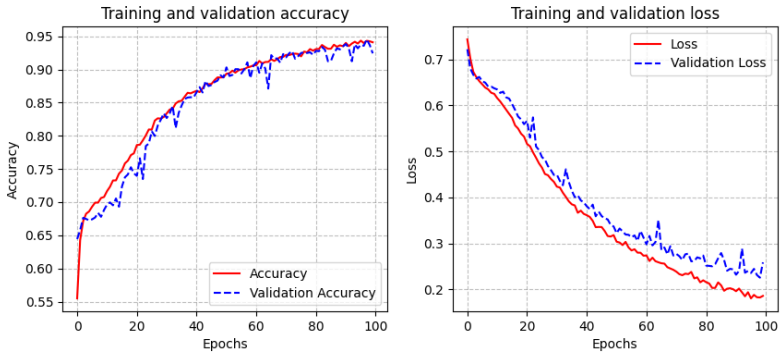


Fig. 7. History of the accuracy and error of the C-BiLSTM architecture applied to the training and validation data for the 50 to 100-word category

Table 3. Results of experiments with texts between 100 and 200 words

-	Accuracy	Loss	AUC	f1_score
MLP	0.95	0.16	0.95	0.98
CNN	0.97	0.10	0.97	0.97
C-LSTM	0.95	0.19	0.95	0.95
C-BiLSTM	0.98	0.12	0.98	0.97

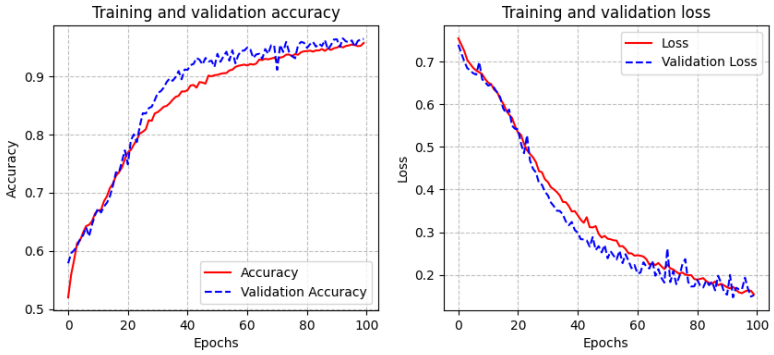


Fig. 8. History of the accuracy and error of the C-BiLSTM architecture applied to the training and validation data for the 100 to 200-word category

6 Conclusion and Future Works

The main objective of this work was to apply Deep Learning architectures to create a prediction methodology to be used with students of EAD (distance education) courses at UFSC, to predict whether students either passed or failed from the analysis of the texts typed by them. Such an approach could be used to anticipate and avoid one of the main causes that lead students to drop out, thus reducing the intrinsic costs of implementing an EAD course and avoiding the waste of valuable resources. The best results obtained for the classification of texts were when applied to the category of texts between 100 and 200 words using the C-BiLSTM Deep Learning architecture, reaching an accuracy of 98%. The same algorithms were also applied in the ranges between 10 and 50 words and 50 and 100 words, obtaining slightly lower results. In some cases, even using a small number of Learning Rates and also a reduced number of epochs, the learnings diverged, preventing obtaining the necessary pattern to make the predictions. This occurred when applying the LSTM architecture to the texts. A point that can also be highlighted is that when we analyze records with a greater number of words the performance of the applied network architectures increases. This fact may, empirically, suggest that it is more accurate to predict the failure and passing of students from the texts with the highest number of words. This fact can be explored in more detail in future works. A possible direction, if sufficient data is available, is the application of the architectures tested here for

the direct prediction of dropout rates. Other points that can be addressed in future work are the application of other Deep Learning architectures as well as direct application without the segregation of texts into word number categories.

References

1. ABED – ASSOCIAÇÃO BRASILEIRA DE EDUCAÇÃO A DISTÂNCIA. Relatório Analítico da Aprendizagem a Distância no Brasil. São Paulo, 2018. http://www.abed.org.br/arquivos/CENSO_EAD_BR_2018_digital_portugues.pdf. Last accessed 22 dec 2021
2. Cislaghi, Renato and others: Um modelo de sistema de gestão do conhecimento em um framework para a promoção da permanência discente no ensino de graduação (2008)
3. Gang Liu and Jiabao Guo: Bidirectional LSTM with attention mechanism and convolutional layer for text classification. Journal: Neurocomputing. Pages: 325-338 (2019)
4. Daniel W. Otter and Julian R. Medina and Jugal K. Kalita: A survey of the usages of deep learning for natural language processing. IEEE Transactions on Neural Networks and Learning Systems, v. 32, n. 2, p. 604-624 (2020)
5. Ian Goodfellow and Yoshua Bengio and Aaron Courville: Deep Learning. Publisher, Location: MIT Press http://www.deeplearningbook.org (2016)
6. Mukesh Kumar, A.J. Singh, Disha Handa: Literature Survey on Educational Dropout Prediction, International Journal of Education and Management Engineering (IJEME), Vol.7, No.2, pp.8-19, 2017.DOI: 10.5815/ijeme (2017)
7. MDUMA, Neema; KALEGELE, Khamisi; MACHUVE, Dina: A Survey of Machine Learning Approaches and Techniques for student dropout prediction (2019).
8. Shahiri, A. M., and Husain, W.: A Review on Predicting Student's Performance Using Data Mining Techniques. Procedia Computer Science, 72, 414-422 (2015)
9. Arsad, P. M., and Buniyamin, N.: A Neural Network Student's Performance Prediction Model (NNSPPM). In 2013 IEEE International Conference on Smart Instrumentation, Measurement and Applications (ICSIMA) (pp. 1-5). IEEE. (2013, November)
10. Natek, S., and Zwilling, M.: Student Data Mining Solution–knowledge Management System Related to Higher Education Institutions. Expert systems with applications, 41(14), 6400-6407 (2014)
11. Wang, Shirui; Zhou, Wenan; Jinag, Chao.: A Survey of Word Embeddings Based on Deep Learning. Computing, v. 102, n. 3, p. 717-740 (2020).
12. Zhong, Junmei; LI, William.: Detecting Customer Churn Signals for Telecommunication Industry Through Analyzing Phone Call Transcripts with Recurrent Neural Networks. In: MLDM (1). p. 93-103. (2019)
13. Mikolov, T., Chen, K., Corrado, G., and Dean, J.: Efficient Estimation of Word Representations in Vector Space. arXiv preprint arXiv:1301.3781. (2013)
14. Hartmann, N., Fonseca, E., Shulby, C., Treviso, M., Rodrigues, J., and Aluisio, S.: Portuguese word embeddings: Evaluating on word analogies and natural language tasks. arXiv preprint arXiv:1708.06025. (2017)
15. Hochreiter, S., and Schmidhuber, J.: Long short-term memory. Neural computation, 9(8), 1735-1780. (1997)
16. Graves, A., and Schmidhuber, J.:. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. Neural networks, 18(5-6), 602-610. (2005)

17. Funahashi, K. I., and Nakamura, Y.: Approximation of dynamical systems by continuous time recurrent neural networks. Neural networks, 6(6), 801-806. (1993)
18. Krizhevsky, A., Sutskever, I., and Hinton, G. E.: Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems, 25, 1097-1105. (2012)
19. Waheed, H., Hassan, S. U., Aljohani, N. R., Hardman, J., Alelyani, S., and Nawaz, R.: Predicting academic performance of students from VLE big data using deep learning models. Computers in Human behavior, 104, 106189. (2020).
20. Hassan, S. U., Waheed, H., Aljohani, N. R., Ali, M., Ventura, S., and Herrera, F.: Virtual learning environment to predict withdrawal by leveraging deep learning. International Journal of Intelligent Systems, 34(8), 1935–1952. (2019).
21. Wasif, M., Waheed, H., Aljohani, N. R., and Hassan, S.-U. Understanding Student Learning Behavior and Predicting Their Performance. In Cognitive Computing in Technology-Enhanced Learning (pp. 1–28). IGI Global. (2019).
22. Marbouti, M. F., and Diefes-Dux, H. A.: Building course-specific regression-based models to identify at-risk students. Age, 26, 1. (2015).
23. Marbouti, F., Diefes-Dux, H. A., and Madhavan, K.: Models for early prediction of at-risk students in a course using standards-based grading. Computers and Education, 103, 1–15. (2016).
24. Hussain, M., Zhu, W., Zhang, W., and Abidi, S. M. R.: Student Engagement Predictions in an e-Learning System and Their Impact on Student Course Assessment Scores. Computational Intelligence and Neuroscience. (2018).
25. Jung, Y., and Lee, J.: Learning engagement and persistence in massive open online courses (MOOCS). Computers and Education, 122, 9–22. (2018).

KCLPruning: A novel Clustering Method on Convolutional Kernels to Accelerate Deep Convolutional Neural Networks

Dana Alqemlas[0000-0002-7894-5309] and Mohammad Jeragh

[1] Kuwait University, Abdullah Al Mubarak Al Sabah 6WXC+X25, Kuwait
[2] Kuwait University, Abdullah Al Mubarak Al Sabah 6WXC+X25, Kuwait

Abstract. In recent years, deep neural networks (DNNs) have advanced to become the state-of-the-art for machine learning (ML). Among the numerous deep learning algorithms, the most well-known is the convolutional neural network (CNN), a type of neural network that has been the leading technique for computer vision tasks. However, deploying convolutional neural networks in resource-constrained devices is challenging due to the model's large number of parameters. As a result, there is a growing interest in model optimization through model acceleration and compression techniques, including network pruning, quantization, and model distillation. However, network pruning is amongst the most advantageous methods to solve the problem as it considerably reduces memory size and bandwidth. This paper proposes a novel method to compress and accelerate neural networks from a small set of spatial convolution kernels. Furthermore, the paper introduces a novel pruning algorithm based on density-based and grid-based clustering methods due to the limited research on clustering-based pruning. Finally, the performance of the pruning algorithm was analyzed, and the experimental results show that the proposed algorithm achieves higher accuracy on image classification than the original model.
Keywords: Deep Learning, Convolutional Neural Networks, Model Compression and Acceleration, Neural Network Pruning, Clustering Methods.

1 Introduction

Deep neural networks (DNNs) have led to the evolution of numerous artificial intelligence (AI) applications, including computer vision [1], speech recognition [2], natural language processing [3], audio recognition [4], and machine translation [5]. The DNNs provide state-of-the-art accuracy for many AI applications, especially computer vision. Computer vision's primary goal is to replicate the human visual system enabling computers to identify objects in images and videos. The field has progressed with the advancement of DNNs as it has exceeded humans in particular image detection and classification tasks.

The development of convolutional neural networks (CNNs) has resulted in exceptional levels of accuracy in computer vision tasks [6]. The CNN models have achieved the best results in image classification [7], object detection [8], and instance segmentation [9]. The success of CNNs is due to the large number of parameters as models

become larger to improve performance. However, deploying these models in resource-constrained devices is challenging due to the model's large number of parameters.
There are several approaches to compress and accelerate deep neural networks, such as parameter pruning [10, 11] and quantization [12], and knowledge distillation [13]. Firstly, the parameter and quantization methods investigate the redundancy in models and remove the redundant model parameters. Furthermore, the knowledge-distillation-based methods learn a distilled model and then train a smaller neural network to replicate the output of a bigger network.

In this paper, we address the problem of compressing and accelerating deep neural networks. Many recent works have explored different techniques to prune deep learning models, such as magnitude-based [14–17], clustering and similarity-based [18–23], and sensitivity analysis-based methods [24–29]. However, there is limited research in pruning based on clustering techniques.

Subsequently, our proposed solution investigates more complex clustering algorithms based on density-based and grid-based clustering methods. Furthermore, the pruning algorithms results were compared through the accuracy, FLOPs, and compression ratio benchmarks. In particular, our compressed models achieve higher accuracy on image classification than the original model.

The main contributions of this paper are to (1) present a comprehensive review of the network compression and acceleration techniques, including network pruning. (2) Propose a novel pruning algorithm based on the CLIQUE clustering method that identifies and removes redundancy in CNN's while maintaining the accuracy and throughput tradeoff. (3) Compare the pruning algorithms results through the accuracy, FLOPs, and compression ratio benchmarks.

The remainder of this research is organized as follows. In Chapter 2, we present work related to model compression in deep neural networks. Chapter 3 describes the research design and implementation. Chapter 4 reports the experimental results and analysis of the pruning algorithm. Finally, conclusions and future work are presented in Chapter 5. 5.

2 Related Work

Numerous studies have been conducted to compress and accelerate neural networks using techniques such as network pruning, quantization, low-rank factorization, and knowledge distillation. However, network pruning is the most famous network optimization technique that compresses the network by removing parameters. Pruning deep neural networks was initially proposed in the 1990s [24, 30]; however, there has been an increasing interest in the field due to the advancement of deep learning. The pruning algorithms can be categorized based on magnitude-based pruning methods, clustering and similarity methods, and sensitivity analysis methods [31]. However, according to the "Methods for Pruning Deep Neural Networks" survey [31], there is limited research in pruning algorithms based on clustering. Consequently, in this paper, we propose to experiment pruning CNN models using the CLIQUE clustering method.

Neural networks generally contain duplicate parameters that can be removed through clustering methods. A few existing works have pruned CNNs based on clustering methods such as K-means clustering, agglomerative hierarchical clustering, etc. Ayinde et al. [32] propose a method of identifying different filters using clustering methods. The similar filters are grouped using an agglomerative hierarchical clustering method, and then one filter is selected from each cluster, and the remaining are removed from the network. The agglomerative hierarchical clustering method operates in a bottom-up" manner as in each step, and the similar clusters are combined into a new cluster. Their results show that their approach provides a better pruning rate and accuracy than other methods, such as network slimming and try-and-learn methods.

Moreover, the authors in [33] introduce a compression algorithm based on corset interpretations of filters. The compression method is based on corset extraction as a smaller set of points is used to represent a large set of points while preserving the necessary property of the initial set. Hence, the sets to be represented are the weight matrices, and the property to be maintained is the activation pattern. Furthermore, the authors utilize the Sparse principal component analysis (SPCA) algorithm, which reduces the number of components while minimizing the reformation error. The SPCA algorithm uses sparsity constraints in the PCA objective. Also, they developed the algorithm with an activation-weighted importance score for each convolutional filter. The method is fast and straightforward as it achieves compression without retraining. The results achieved by this method reduced the size of AlexNet by approximately ten times without decreasing the accuracy. However, there are still redundancies remaining in these compact networks, and the matrix decomposition complexity may be higher for more extensive networks

Furthermore, the work in [34] proposes compressing CNNs by reforming the network using a small collection of spatial convolution kernels. First, the 2D kernel centroids are extracted using K-means clustering before training. Afterward, each centroid replaces the corresponding kernels of the same cluster. Then the indexed representation is saved as an alternative to the whole kernel. The model is then finetuned, and the kernels in each cluster share their weights. The results achieved include 10x compression with clustering and 30x compression with pruning and clustering kernels. On the other hand, they did not study the compression of FC layers and pointwise convolutions, which are common in modern architectures. Also, they did not investigate the relationship of quantization error to accuracy.

Zhou et al. [35] propose a method to prune redundant channels by proposing an additional cluster loss term in the loss function, which forces filters in each cluster to be alike online. This method is useful for pruning networks within residual blocks as the clusters in every layer can be defined; however, it is unchangeable with different models. Additionally, the work [36] prunes filters based on the K-means algorithm. They obtain the relationship of similarity among filters and keep the filters close to the centroid of each cluster.

Also, the authors in [37] propose a feature-agnostic method, which obtains a relationship between input feature maps and kernels. The filters are pruned based on a kernel sparsity and entropy (KSE) indicator. It can compress each layer efficiently, providing fast performance, achieving 5x FLOPs reduction and 3x compression on ResNet-

50. However, the issue with the filter pruning methods is that each layer's compression ratio must be manually set, which is time-consuming. In addition, Centripetal SGD [38] compresses the CNN model by clustering the identical filters. The method groups the filters through K-means and makes the filters in one cluster the same through backpropagation to place identical filters into one cluster. Yu et al. [39]demonstrate an efficient structural pruning method, Kernel Cluster Pruning (KCP), that clusters similar filters based on a technique inspired by the K-means clustering approach. Their method performs better than several state-of-the-art pruning methods with minor accuracy loss.

Most of the recent pruning algorithms based on clustering focus on the K-means clustering approach; however, numerous other approaches can be explored. Therefore, we experiment with a more complex clustering method, CLIQUE.

3 Network Pruning

The pruning section begins with introducing the CNN formulation and pruning algorithm. Afterward, we present the algorithms based on the partitional-based, density-based, and grid-based clustering.

3.1 CNN Formulation

We present the methodology to be followed to formulate the pruning algorithm that compresses convolutional neural networks (CNNs) by reconstructing the network from a small set of spatial convolution filters. The terms and notations are defined as follows; we will assume the N filters have the same spatial sizes for simplicity. Also, the convolutional layer m comprises of the input x and output y with the C_{in} and C_{out} channels. Then, w^k is the weight of the k convolutional layer where $w^k \in R^{C_{out} \times C_{in} \times h \times w}$. Furthermore, the ith input channel of x is denoted as x_i and jth output channel of y is denoted as y_j. The output of the channel y_j is computed as Eq 1.

$$y_j = \sum_{i=1}^{C_{in}} w_{ij} * x \tag{1}$$

The CNN models are generally trained without any structural limits on the shape of weight tensors. As a result, there may be N individual filters in the network. Hence, our primary goal is to represent the filters more efficiently by means of the K-means and CLIQUE clustering approaches.

3.2 Pruning Algorithm

The general pruning algorithm is detailed in algorithm 1, with the primary goal of pruning the filters in the CNN model to remove the redundant filters [40]. The algorithm removes the filters using either of the K-means and CLIQUE clustering approaches. We start by going through the layers of the CNN model, and then we obtain the different filters W_c. Then, the dimensionality reduction technique Principal Component Analysis (PCA) is applied to reduce the number of dimensions. Furthermore, the constancy of the optimal hyperparameters is identified through the elbow method. The technique

begins by choosing a cluster size between 1 and N, where N is less than or equal to the number of points in the dataset. After that, the sum of intracluster distances is computed for each cluster. The elbow point is then identified at which the rate of the sums drops slowly. The elbow method is often used for the K-means algorithm; however, it may be applied to various clustering methods. Accordingly, the elbow rule was utilized in our pruning approach to determine the optimal number of clusters for the K-means and the optimal unit size for the CLIQUE. Afterward, the function CLUSTERING will provide the clustered filters n_f and indices of clusters $L_{f,}$ which corresponds to a fraction of the indices of columns W_c. After that, a smaller filter matrix is produced $W_{cpruned}$ that is created using the L_f to subgroup W_c.

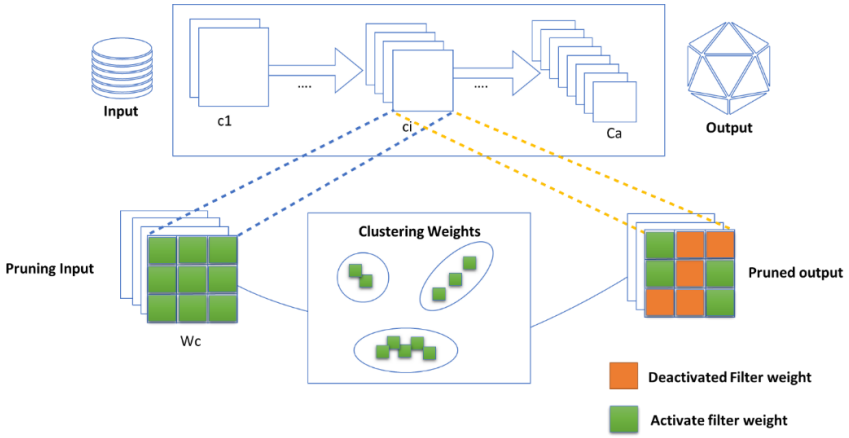


Fig. 1. Illustration of the Proposed Pruning Method

Algorithm 1: Filter-based Pruning with Automated Hyperparameter Selection

1. **input**: {a←number of layers, m ← layer, W ← weight of layer m, c ←0}
2. **while** c less than a
3. **get**: convolutional filters of layer c
4. **obtain**: different filters W_c
5. **perform** PCA on filters W_c
6. **define** SSE Error List
4. OH = 1
5. **set** Hmax
6. **while** OH less than Hmax
7. CLUSTERING(Wc, OH)
8. **calculate** cluster SSE error
9. **append** SSE error to SSE Error List
10. OH ← OH + 1
11. **end while**
12. **plot** elbow plot with Hmax versus SSE Error List
13. **find** H (optimal hyperparameter) by using knee locator
14. L_f, *nf* = **CLUSTERING** (W_c, H)
8. **define** W_c^{pruned}
9. k ← 0
10 **while** length of L_f is greater than k

11. **copy** $L_{f(k)}$ column into W_c into k column W_c^{pruned}
12. k ← k+1
13. **end while**
14. c ← c +1
15. **end while**
16. **build:** pruned model
17. **initialize**: weights of the c layer with W_c^{pruned}

3.3 K-Means Clustering Approach

The K-means clustering approach is one of the earliest and well-researched literature clustering techniques. The K-means clustering technique is based the partitional clustering. First, let us define the terms and assumptions used in this section and the subsequent sections. The filters h*h mentioned in the previous section may be flattened out into a feature 2d matrix N x h^2. Assuming the filters are grouped into k clusters using the K-means clustering technique. The technique assigns each entity to the closest centroid of the cluster it belongs to in a multi-dimensional space.

Each filter may then be replaced by the centroid value of such cluster and reshaped back to h*h. This would significantly reduce the total number of filters from N to k and compress the model size. The accurate value of k may be derived from multiple experiments on the dataset and is also a function of the tradeoff between size, speed of inference, and accuracy.

The first approach to be implemented for the clustering of the filters is the K-means approach which is detailed in algorithm 2 – K-Means approach below. The algorithm's input is the weight of the filters and the number of desired clusters. The centroids are then initialized, and the K data points are selected for the centroids while there is no convergence of the criteria. Afterward, the data points are assigned to the closest centroid. Then each data point is assigned to the closest cluster. Later, the centroids are computed for the clusters by selecting the mean of the data points that belong to each cluster [41].

Algorithm 2: CLUSTERING – Adaptive K-Means Clustering Approach

1. function **CLUSTERING():**
2. **input**: {**W ← weight of filters, K ← number of desired clusters**}
3. **while** there is no convergence of the criteria
4. **define** centroid initial values for n_1, n_2, … n_k
5. **for** i = 1 … N
6. **find** closest center n_k for each w_i (data point)
7. **update** n_k with the value of w_i (data point)
8. **end for**
9. **calculate:** centroid values as mean of data points $w_{i...n}$ assigned to the relevant cluster
10. **for** i = 1 ….k
11. **calculate:** centroid value as mean of data points $w_{i...n}$ assigned to the cluster L_f
13. **set** n_i to be the new centroid
14. **end for**
15. *nf* ← K
16. **end while**
17. return L_f, *nf*
18. **end function**

3.4 CLIQUE Clustering Approach

The second approach for the pruning algorithm is based on the CLIQUE (Clustering in QUEst) clustering technique [42]. The CLIQUE algorithm is based on density-based and grid-based clustering. The algorithm obtains the clusters in subspaces of high dimensional data using the number of grids and density threshold input parameters.

The CLIQUE clustering technique includes three main steps: recognizing the subspaces that include clusters, recognizing the clusters, and creating the minimal description for the clusters. CLIQUE first divides the dimensions into grids and then obtains the dense regions depending on the threshold value. The clusters are then generated from the dense subspaces utilizing the Apriori approach. The Apriori approach states that a cluster in a k-dimensional space will also be part of a cluster in any (k-1) dimensional projections of this space. Afterward, the CLIQUE algorithm produces minimal descriptions for the clusters by determining the maximal regions that include a cluster of associated dense units for each cluster and the minimum cover for each cluster.

In this approach, the filters will be clustered based on density and grid based approaches which is detailed in algorithm 3 – CLIQUE approach. First of all, the input consists of the weight of the filters defined in a vector of n-dimensional points W = $\{w_1, w_2, \dots, w_n\}$ where $w_i = \{w_{i1}, w_{i2}, \dots, w_{in}\}$. The size of the units (U) and density threshold (φ) input parameters were selected based on the elbow method. The dimensions are first divided into equal intervals b according to the unit size U. Then, the dense spaces are identified through a bottom-up algorithm that prunes the search space by utilizing the monotonicity of the cluster with regard to its dimension through the Apriori approach. The partitions p are considered dense if they are greater than the density threshold φ. The dense dimensions are joined to the next dimension, and the Apriori property discards the dense units from the ser D_k of dimensions that have a projection in (k-1) dimensions that are not in D_{k-1}. Afterward, the clusters are identified using the depth-first search algorithm to find the connected components. Lastly, the minimal description of the clusters is generated by checking both directions of the i-th dimension, attempting to cover every unit in C [43].

Algorithm 3: CLUSTERING – Adaptive CLIQUE Clustering Approach

```
1.  function CLUSTERING(Wc, H):
2.  input: {W ← weight of filters (data) , H ← size of units, φ ← density threshold }
3.  p ←number of dimensions
3.  for each dimension p in W
4.      B ← partition p into equal intervals according to H
5.      b ← single interval
6.      fix a minimum input parameter ∅ (by applying Apriori property)
7.      for each b in B
8.          if density of b > φ
9.          D1 ← D1 ∪ b
10.     end for
11. end for
12. K ← 1  // dimension we start with
13. while D_K != ∅
14.         K ← K + 1
15.         D_K ← set of k-dimensional dense units were (k-1) ∈ D_{K-1}
16. end while
```

```
17. set m ← number of clusters
18.  for each high coverage subspace S
19.      take set of dense units D_p in S
20.      while D_p != ∅
21.            m ← 1
22.            select randomly chosen unit u from D_p
23.            Cm ← current cluster = u ∪ {v; for all v ∈ D_p connected to u}
24.            D_p ← D_p – Cm
25.            m ← m+1
26.      end while
27. end for
28.  for each cluster Cm in C
29.      while Cm != ∅
30.           I ← set containing all the units covered in Cm
31.           choose a dense unit u in Cm
32.           for i = 1 to L
33.           I ← I ∪ dense unit of Cm in the neighborhood of u
34.           end for
35.           Cm ←Cm-1
36.       end while
37. end for
38. define L_f as Cm
39. define nf  as m
40. return L_f, nf
41. end function
```

4 Experiments

The efficiency of our clustering-based pruning approach is demonstrated experimentally using the two representatives CNN architectures ResNet and VGG pretrained on the CIFAR dataset loaded from torchvision.models of the PyTorch ML framework [44]. The pruning algorithms based on K-means and CLIQUE clustering approaches were compared, applying them to all layers of our neural networks and measuring the resulting accuracy and complexity of the pruned model. Moreover, we quantified the model's performance and accuracy using the sparsity and model accuracy metrics. First, we set the K-means and CLIQUE pruning parameters for each layer and prune an equal number of filters to achieve comparable results. Furthermore, the CLIQUE's grid-size parameter and the cluster for each layer are defined. Afterward, the number of clusters in the models is specified by the CLIQUE clustering approaches. Following that, we compare two approaches based on the sparsity and accuracy of the test set.

There is a tradeoff between the model sparsity and prediction accuracy as the more parameters that are removed, the less accurate the prediction becomes. Furthermore, the K-means algorithm consistently generates better results on the CIFAR-10 dataset, as shown by the results in table 1. This is because in K-means, the pruned layer is made up of centroids, and all other filters are removed from the network, but in CLIQUE, a random filter from each cluster is selected, and all other filters are deleted. Similarly,

as demonstrated in table 2, K-means deliver better results in the initial experiments on the VGG-16 network.

Nevertheless, the pruning algorithm based on the K-means and CLIQUE clustering approaches requires further enhancements introduced in the following sections. As firstly, the hyperparameters for the clustering algorithms for each layer must be tuned in advance. Additionally, the CLIQUE algorithm requires a more accurate cluster approximation as it currently uses a random filter to represent a cluster.

Table 1. Pruning results of ResNet-56 on CIFAR-10.

Clustering Method	Clustering Parameters	Resulting Number of Filters	Number of Model Parameters	Accuracy	Sparsity
None	--	--	2.70E+05	80%	0%
CLIQUE	[8, 8, 8, 16, 16, 16, 32, 32, 32]	[9, 6, 5, 16, 20, 20, 46, 40, 43]	1.73E+05	58%	64.2%
K-Means	[9, 6, 5, 16, 20, 20, 46, 40, 43]	[9, 6, 5, 16, 20, 20, 46, 40, 43]	1.73E+05	65%	64.2%
CLIQUE	[20, 20, 20, 40, 40, 40, 80, 80, 80]	[15, 12, 11, 25, 29, 29, 62, 62, 59]	2.51E+05	77%	92.9%
K-Means	[15, 12, 11, 25, 29, 29, 62, 62, 59]	[15, 12, 11, 25, 29, 29, 62, 62, 59]	2.51E+05	78%	92.9%
CLIQUE	[(24, 24, 24, 48, 48, (48, 0), (96, 0), (96, 0), (96, 0)]	[14, 14, 10, 27, 31, 31, 64, 61, 62]	2.58E+05	79%	95.6%
K-Means	[14, 14, 10, 27, 31, 31, 64, 61, 62]	[14, 14, 10, 27, 31, 31, 64, 61, 62]	2.58E+05	80%	95.6%
CLIQUE	[32, 32, 32, 64, 64, 64, 128, 128, 128]	[16, 15, 14, 28, 31, 32, 63, 62, 62]	2.61E+05	79%	96.8%
K-Means	[16, 15, 14, 28, 31, 32, 63, 62, 62]	[16, 15, 14, 28, 31, 32, 63, 62, 62]	2.61E+05	79%	96.8%
CLIQUE	[36, 36, 36, 72, 72, 72, 144, 144, 144]	[16, 15, 15, 29, 31, 31, 64, 63, 62]	2.63E+05	79%	97.6%
K-Means	[16, 15, 15, 29, 31, 31, 64, 63, 62]	[16, 15, 15, 29, 31, 31, 64, 63, 62]	2.63E+05	80%	97.6%
CLIQUE	[40, 40, 40, 80, 80, 80, 160, 160, 160]	[16, 16, 14, 28, 32, 32, 64, 64, 63]	2.66E+05	79%	98.7%
K-Means	[16, 16, 14, 28, 32, 32, 64, 64, 63]	[16, 16, 14, 28, 32, 32, 64, 64, 63]	2.66E+05	80%	98.7%

Table 2. Pruning results of VGG-16 on CIFAR-10.

Clustering Method	Clustering Parameters	Resulting Number of Filters	Number of Model Parameters	Accuracy	Sparsity
None	--	--	1.47E+07	86%	0%
CLIQUE	[32, 32, 64, 64, 128, 128, 128,128, 128, 128, 128]	[25, 37, 101, 143, 348, 378]	8.33E+06	59%	56.6%
K-Means	[25, 37, 101, 143, 348, 378]	[25, 37, 101, 143, 348, 378]	8.33E+06	67%	56.6%

CLIQUE	[80, 80, 160, 160, 320, 320, 320, 320, 320, 320, 320]	[38, 80, 189, 378, 474, 486]	1.27E+07	81%	86.2%
K-Means	[38, 80, 189, 378, 474, 486]	[38, 80, 189, 378, 474, 486]	1.27E+07	83%	86.2%
CLIQUE	[96, 96, 192, 192, 384, 384, 384, 384, 384, 384, 384]	[49, 92, 211, 413, 485, 497]	1.33E+07	83%	90.3%
K-Means	[49, 92, 211, 413, 485, 497]	[49, 92, 211, 413, 485, 497]	1.33E+07	85%	90.3%
CLIQUE	[128, 128, 256, 256, 512, 512, 512, 512, 512, 512, 512)]	[50, 107, 230, 453, 499, 505]	1.39E+07	84%	94.5%
K-Means	[50, 107, 230, 453, 499, 505]	[50, 107, 230, 453, 499, 505]	1.39E+07	86%	94.5%
CLIQUE	[144, 144, 288, 288, 576, 576, 576, 576, 576, 576, 576]	[54, 108, 236, 463, 500, 507]	1.41E+07	85%	95.4%
K-Means	[54, 108, 236, 463, 500, 507]	[54, 108, 236, 463, 500, 507]	1.41E+07	86%	95.4%
CLIQUE	[(160, 160, 320, 320, 640, 640, 640, 640, 640, 640, 640]	[56, 109, 238, 474, 504, 506]	1.42E+07	85%	96.2%
K-Means	[56, 109, 238, 474, 504, 506]	[56, 109, 238, 474, 504, 506]	1.42E+07	86%	96.2%

4.1 Automated Hyperparameter Selection

To further improve our approach's performance, we automate the hyperparameter selection process, which includes determining the optimal hyperparameters for each layer using the elbow method [45–48]. The elbow method is a visual way for assessing the constancy of the optimal hyperparameters. In order to identify the optimal number of clusters for the K-means and the appropriate unit size for the CLIQUE, we used the elbow rule in our pruning strategy. First, the hyperparameter is initialized, and then the value is incremented. Afterward, the sum of square error results for each value of H is calculated. Lastly, the optimal hyperparameter is identified by comparing the difference SSE of each cluster, with the most intense difference creating the elbow angle, as shown in figure 2. The SSE formula is shown below, where x_j is the object in the cluster c_i and centroid.

$$SSE = \sum_{i=1}^{k} \sum_{x_j \in C_i} \left\| x_j - \mu_i \right\|^2 \quad (2)$$

As a result of our enhancements, the pruning based on the CLIQUE approach achieves better results than the K-means approach, as illustrated in tables 3 and 4, which is a significant improvement over the initial version. The CLIQUE clustering algorithm performed better because it combines the concepts of density-based and grid-based clustering. Also, it can detect clusters in subspaces with drastically varied dimensionalities. Additionally, it introduces the induction of associated rules through the Apriori algorithm. Furthermore, CLIQUE can detect clusters in subspaces of significantly

different dimensionality. Moreover, it utilizes the minimum depth length principle to select appropriate subspaces. In addition, it interprets the clusters in terms of the disjunctive normal form representation.

Table 3. Pruning results of ResNet-56 on CIFAR-10 with Automatic Hyperparameter Selection.

Clustering Method	Number of filters	Number of Model Parameters	Accuracy	Sparsity
None	--	2.70E+05	83%	0%
CLIQUE	[16, 16, 16, 31, 32, 32, 64, 64, 62]	2.67E+05	82%	99%
KMEANS	[16, 16, 16, 32, 32, 32, 40, 44, 44]	2.03E+05	59%	75.2%

Table 4. Pruning results of VGG-16 on CIFAR-10 with Automatic Hyperparameter Selection.

Clustering Method	Number of filters	Number of Model Parameters	Accuracy	Sparsity
None	--	1.47E+07	86%	0%
CLIQUE	[48, 103, 168, 295, 313, 376]	9.62E+06	82%	65.3%
KMEANS	[40, 96, 176, 372, 352, 376]	1.05E+07	82%	71.4%

Table 5. Pruning results of VGG-19 on CIFAR-10 with Automatic Hyperparameter Selection.

Clustering Method	Number of filters	Number of Model Parameters	Accuracy	Sparsity
None	[64, 64, 128, 128, 256, 256, 256, 256, 512, 512, 512, 512, 512, 512, 512, 512]	2.00E+07	84%	0%
CLIQUE	[64, 64, 128, 95, 256, 202, 256, 207, 512, 377, 512, 412, 512, 243, 512, 512]	1.47E+07	80%	73.3%
KMEANS	[64, 44, 128, 84, 256, 176, 256, 180, 512, 376, 512, 376, 512, 368, 512, 512]	1.51E+07	77%	75.5%

4.2 Filter Selection Optimization

This section presents the enhancement of the CLIQUE-based pruning algorithm using the Lp-norm to select the optimal number of filters. The Lp-norm evaluates the importance of each filter by calculating the sum of its absolute weights, as shown in the below equation.

$$\|\boldsymbol{f}\|_{\boldsymbol{p}} = (\sum_{\boldsymbol{i}} |\boldsymbol{f}_{\boldsymbol{i}}|^{\boldsymbol{p}})^{\boldsymbol{1/p}} \quad (3)$$

In our experiments, we compare the L1-norm with L2-norm for filter pruning on the CIFAR dataset. As shown in tables 6 and 7, the L2-norm works slightly better than the L1-norm. Also, we find L1-norm is an efficient heuristic for filter selection due to its data-free nature. Furthermore, the L1-norm will reduce some weights to 0, inducing sparsity in the weights. Moreover, we compare our method with [49, 50] on ResNet-56 as shown in table 8; our method achieves the best results compared to the other pruning methods.

Table 6. Pruning results of ResNet on CIFAR-10 with Filter Selection Optimization.

Clustering Method	Number of filters	Number of Parameters	Sparsity	Accuracy	L1 Norm	L2 Norm
CLIQUE	[16, 16, 16, 31, 32, 32, 64, 62, 62]	2.65E+05	98.1%	79%	79.53%	79.52%
KMEANS	[16, 16, 16, 32, 32, 20, 40, 44, 44]	1.96E+05	72.6%	59%	-	-

Table 7. Pruning results of VGG-16 on CIFAR-10 with Filter Selection Optimization.

Clustering Method	Number of filters	Number of Parameters	Sparsity	Accuracy	L1 Norm	L2 Norm
CLIQUE	[16, 16, 16, 31, 32, 32, 64, 62, 62]	9.63E+06	65.4%	82%	82.85%	82.82%
KMEANS	[16, 16, 16, 32, 32, 20, 40, 44, 44]	1.08E+07	73.5%	82%	-	-

Table 8. Accuracy performance of ResNet-56 on CIFAR-10 under different pruning methods.

Methods	% Accuracy drop	% Parameters pruned
PFEC [49]	1.73	13.70%
FPGM [50]	0.66	38.71%
FPGM [50]	1.55	59.13%
Ours	0.47	98.10%

4.3 Finetuning

Previously, we pruned CNNs by removing unnecessary filters and evaluating the pruned version's accuracy and performance. However, in practice, the pruned network must be retrained and finetuned before being used in production. As a result, in the following experiments, we run multiple training epochs on the pruned networks and evaluate their accuracy following the finetuning process. At this stage, we simulate a more realistic environment using the Places365 dataset.

We finetune the pruned models for all experiments on PLACES365 datasets using the stochastic gradient descent optimizer and a batch size of 20 epochs. The learning rate starts at 0.001, and the momentum decays at a rate of 0.9. As a result, the CLIQUE algorithm consistently generates better results on the PLACES365 dataset, as shown by the results in table 9.

Table 9. Pruning results of Fine-tuned ResNet on PLACES365.

Clustering Method	Number of Model Parame-ters	Spar-sity	top5 pruned	top5 fine-tuned 1ep	top5 fine-tuned 4ep	top5 fine-tuned 8ep	top5 fine-tuned 15ep	top5 fine-tuned 16ep
None	2.43E+07	0	0.849					
CLIQUE	2.11E+07	87%	7.39E-01	0.37	0.55	0.60	0.63	0.86
KMEANS	2.18E+07	90%	5.98E-01	0.34	0.53	0.61	0.62	0.83

5 Conclusion

In this paper, we present a novel pruning method KCLPruning, which identifies and removes redundancy in CNN's using complex clustering algorithms based on density-based, grid-based, and partitional-based clustering methods. The results were further improved using automated hyperparameter selection and filter selection optimization. The methods' validity was evaluated using a variety of CNN architectures and datasets. The networks pruned using our approach achieves better results than state-of-the-art methods. For future work, we will explore more clustering algorithms to compress and accelerate CNNs.

References

1. Cohen TS, Geiger M, Köhler J, Welling M (2018) Spherical cnns. arXiv preprint arXiv:180110130
2. Yu D, Deng L (2016) Automatic Speech Recognition. Springer
3. Devlin J, Chang M-W, Lee K, Toutanova K (2018) Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:181004805
4. Mozer FS, Savoie RE, Teasley WT (2004) Audio recognition peripheral system
5. Tu Z, Lu Z, Liu Y, et al (2016) Modeling coverage for neural machine translation. arXiv preprint arXiv:160104811
6. Redmon J, Farhadi A (2018) Yolov3: An incremental improvement. arXiv preprint arXiv:180402767
7. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp 770–778
8. Everingham M, van Gool L, Williams CKI, et al (2010) The pascal visual object classes (voc) challenge. International journal of computer vision 88:303–338
9. Girshick R, Donahue J, Darrell T, Malik J (2014) Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp 580–587
10. Cheng Y, Wang D, Zhou P, Zhang T (2017) A survey of model compression and acceleration for deep neural networks. arXiv preprint arXiv:171009282

11. Wu J, Leng C, Wang Y, et al (2016) Quantized convolutional neural networks for mobile devices. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp 4820–4828
12. Hanson S, Pratt L (1988) Comparing biases for minimal network construction with back-propagation. Advances in neural information processing systems 1:177–185
13. Hinton G, Vinyals O, Dean J (2015) Distilling the knowledge in a neural network. arXiv preprint arXiv:150302531
14. Weigend AS, Rumelhart DE, Huberman BA (1991) Back-propagation, weight-elimination and time series prediction. In: Connectionist models. Elsevier, pp 105–116
15. Chauvin Y (1988) A Backpropagation Algorithm with Optimal Use of Hidden Units. In: NIPS. pp 519–526
16. Weigend AS, Rumelhart DE, Huberman BA (1991) Generalization by weight-elimination applied to currency exchange rate prediction. In: [Proceedings] 1991 IEEE International Joint Conference on Neural Networks. IEEE, pp 2374–2379
17. Zhou Z, Zhou W, Hong R, Li H (2018) Online filter weakening and pruning for efficient convnets. In: 2018 IEEE International Conference on Multimedia and Expo (ICME). IEEE, pp 1–6
18. Chen AM, Lu H, Hecht-Nielsen R (1993) On the geometry of feedforward neural network error surfaces. Neural computation 5:910–927
19. Han S, Liu X, Mao H, et al (2016) EIE: Efficient inference engine on compressed deep neural network. ACM SIGARCH Computer Architecture News 44:243–254
20. Li L, Zhu J, Sun M-T (2019) Deep learning based method for pruning deep neural networks. In: 2019 IEEE International Conference on Multimedia & Expo Workshops (ICMEW). IEEE, pp 312–317
21. RoyChowdhury A, Sharma P, Learned-Miller E, Roy A (2017) Reducing duplicate filters in deep neural networks. In: NIPS workshop on Deep Learning: Bridging Theory and Practice. p 1
22. Sussmann HJ (1992) Uniqueness of the weights for minimal feedforward nets with a given input-output map. Neural networks 5:589–593
23. Zhou Z, Zhou W, Li H, Hong R (2018) Online filter clustering and pruning for efficient convnets. In: 2018 25th IEEE International Conference on Image Processing (ICIP). IEEE, pp 11–15
24. Mozer MC, Smolensky P (1989) Skeletonization: A technique for trimming the fat from a network via relevance assessment. In: Advances in neural information processing systems. pp 107–115
25. LeCun Y, Denker JS, Solla SA (1990) Optimal brain damage. In: Advances in neural information processing systems. pp 598–605
26. Hassibi B, Stork DG, Wolff GJ (1993) Optimal brain surgeon and general network pruning. In: IEEE international conference on neural networks. IEEE, pp 293–299
27. Cohen JP, Lo HZ, Ding W (2016) RandomOut: Using a convolutional gradient norm to rescue convolutional filters. arXiv preprint arXiv:160205931
28. Lee N, Ajanthan T, Torr PHS (2018) Snip: Single-shot network pruning based on connection sensitivity. arXiv preprint arXiv:181002340
29. Lin S, Ji R, Li Y, et al (2018) Accelerating Convolutional Networks via Global & Dynamic Filter Pruning. In: IJCAI. p 8
30. Kruschke JK (1988) Creating local and distributed bottlenecks in hidden layers of backpropagation networks. Proceedings 1998 Connectionist Models Summer School 120–126
31. Vadera S, Ameen S (2020) Methods for Pruning Deep Neural Networks. arXiv preprint arXiv:201100241

32. Ayinde BO, Inanc T, Zurada JM (2019) Redundant feature pruning for accelerated inference in deep neural networks. Neural Networks 118:148–158
33. Dubey A, Chatterjee M, Ahuja N (2018) Coreset-based neural network compression. In: Proceedings of the European Conference on Computer Vision (ECCV). pp 454–470
34. Son S, Nah S, Lee KM (2018) Clustering convolutional kernels to compress deep neural networks. In: Proceedings of the European Conference on Computer Vision (ECCV). pp 216–232
35. Zhou Z, Zhou W, Li H, Hong R (2018) Online filter clustering and pruning for efficient convnets. In: 2018 25th IEEE International Conference on Image Processing (ICIP). IEEE, pp 11–15
36. Li L, Zhu J, Sun M-T (2019) Deep learning based method for pruning deep neural networks. In: 2019 IEEE International Conference on Multimedia & Expo Workshops (ICMEW). IEEE, pp 312–317
37. Li Y, Lin S, Zhang B, et al (2019) Exploiting kernel sparsity and entropy for interpretable CNN compression. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp 2800–2809
38. Ding X, Ding G, Guo Y, Han J (2019) Centripetal sgd for pruning very deep convolutional networks with complicated structure. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp 4943–4953
39. Yu P-H, Wu S-S, Chen L-G (2021) KCP: Kernel Cluster Pruning for Dense Labeling Neural Networks. arXiv preprint arXiv:210106686
40. Han S, Mao H, Dally WJ (2015) Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. arXiv preprint arXiv:151000149
41. Mohd WMW, Beg AH, Herawan T, Rabbi KF (2012) MaxD K-means: A clustering algorithm for Auto-generation of centroids and distance of data points in clusters. In: International Symposium on Intelligence Computation and Applications. Springer, pp 192–199
42. Agrawal R, Gehrke J, Gunopulos D, Raghavan P (2005) Automatic subspace clustering of high dimensional data. Data Mining and Knowledge Discovery 11:5–33
43. Hu T, Liu C, Tang Y, et al (2014) High-dimensional clustering: a clique-based hypergraph partitioning framework. Knowledge and information systems 39:61–88
44. pytorch TORCHVISION.MODELS
45. López-Rubio E, Palomo EJ, Ortega-Zamorano F (2018) Unsupervised learning by cluster quality optimization. Information Sciences 436:31–55
46. Bholowalia P, Kumar A (2014) EBK-means: A clustering technique based on elbow method and k-means in WSN. International Journal of Computer Applications 105:
47. Syakur MA, Khotimah BK, Rochman EMS, Satoto BD (2018) Integration k-means clustering method and elbow method for identification of the best customer profile cluster. In: IOP Conference Series: Materials Science and Engineering. IOP Publishing, p 012017
48. Kodinariya TM, Makwana PR (2013) Review on determining number of Cluster in K-Means Clustering. International Journal 1:90–95
49. Li H, Kadav A, Durdanovic I, et al (2016) Pruning filters for efficient convnets. arXiv preprint arXiv:160808710
50. He Y, Liu P, Wang Z, et al (2019) Filter pruning via geometric median for deep convolutional neural networks acceleration. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp 4340–4349

Ammunition Component Classification Using Deep Learning

Hadi Ghahremannezhad[1], Chengjun Liu[1], and Hang Shi[2]

[1] New Jersey Institute of Technology
Department of Computer Science
Newark, NJ 07102 USA
[2] Innovative AI Technologies
Newark, NJ 07201, USA

Abstract. Ammunition scrap inspection is an essential step in the process of recycling the ammunition metal scrap. Most ammunition is composed of a number of components, including case, primer, powder, and projectile. Ammo scrap containing energetics is considered to be potentially dangerous and should be separated before the recycling process. Manually inspecting each piece of scrap is tedious and time-consuming. We have gathered a dataset of ammunition components with the goal of applying artificial intelligence for classifying the safe and unsafe scrap pieces automatically. First, two training datasets are manually created from visual and x-ray images of ammo. Second, the x-ray dataset is augmented using the spatial transforms of histogram equalization, averaging, sharpening, power law, and Gaussian blurring in order to compensate for the lack of sufficient training data. Lastly, the representative YOLOv4 object detection method is applied to detect the ammo components and classify the scrap pieces into safe and unsafe classes, respectively. The trained models are tested against unseen data in order to evaluate the performance of the applied method. The experiments demonstrate the feasibility of ammo component detection and classification using deep learning. The datasets and the pre-trained models are available at https://github.com/hadi-ghnd/Scrap-Classification.

Keywords: ammunition component classification, computer vision, dataset, deep learning, YOLO, YOLOv4.

1 Introduction

The non-usable ammunition goes through a rotary kiln incinerator (RKI) before recycling. The safeness of the ammunition scrap should be confirmed before the process of recycling. If the scrap still contains a considerable amount of energetics after the incineration process they are considered to be potentially dangerous. Therefore, the scrap pieces are inspected in order to make sure there is no energetics left. Manual inspection is a laborious, inaccurate, costly, and time-consuming step and is prone to human error. Hence, there is a need for a reliable and effective method to inspect the ammunition scrap automatically. Visual

imaging and x-ray penetration are beneficial in detecting and discriminating the energetics remaining in the ammo scrap.

In this study, we have generated two datasets of visual and x-ray ammo images that are used for training a deep convolutional neural network (DCNN) to aid with the detection of explosive hazards on metallic ammunition scrap. The goal is to sort the pure metal scrap from the scrap pieces that contain traces of explosive hazards. The two classes are named MDAS (Material Documented as Safe) and MPPEH (Material Potentially Possessing Explosive Hazard), respectively. Due to the lack of a sufficient number of x-ray images, several data augmentation techniques are applied as a pre-processing step. The representative YOLOv4 object detection method [2] is applied in order to train two DCNN models against the gathered training data. The trained models are evaluated against the testing datasets according to appropriate measures to verify the effectiveness of the applied approach.

The remainder of this paper is organized as follows. Section 2 the required background material about DCNNs and the YOLOv4 method is briefly described. Section 3 details the dataset generation and algorithms applied for training the deep CNN models that are used for ammunition component detection and explains the criteria used in the classification process. In section 4 experiments conducted for evaluating the trained models are explained along with the assessment measures used to analyze the performance. Finally, the paper is concluded in section 5.

2 Background

Object detection and classification is one of the fundamental steps in many applications of computer vision and video analytics, such as robot vision [19,13], autonomous driving [18,6,7], and traffic monitoring [15,9,5,10,12,8,25,11,24]. Throughout the previous years many studies have been published that apply statistical methods, such as Support Vector Machine (SVM) [20], efficient SVM (eSVM) [4], Adaboost [26], the Bayesian Discriminating Features (BDF) method [14], and discriminant analysis [3].

In recent years, deep convolutional neural networks have been popularly used for many tasks in computer vision, including object detection [16,27]. A Convolutional Neural Network (CNN) is a deep learning algorithm that operates on an image as the input in order to train several parameters and extract high-level features for various tasks. The architecture of Convolutional Neural Networks (CNNs) is well-suited to images in that it captures the spatial and temporal dependencies by applying appropriate filters. The convolution layers in a CNN are usually coupled with activation functions and pooling layers for increasing the non-linearity and size reduction.

One of the most representative object detection approaches in the You Only Look Once (YOLO) deep learning method was introduced in 2015 [21]. The core idea of this method is to divide the input image into an $S \times S$ grid where each cell of the grid either represents the background class or is used for the detection

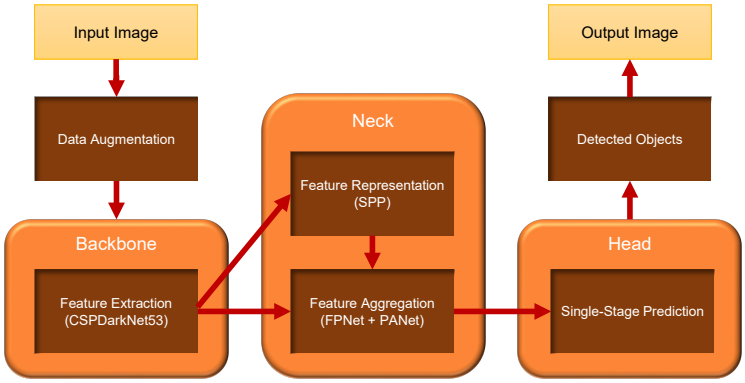


Fig. 1. The main modules in the YOLOv4 model.

of the object the center of which falls in that cell. At each cell, a predefined number (B) of bounding boxes are generated along with their corresponding confidence scores, which indicates how likely the box is to contain an object. The probability of each object is multiplied with the intersection over union (IOU) of the predicted bounding box and the ground truth box to calculate the confidence scores. As many object detection methods, YOLO utilizes the non-maximum suppression algorithm to remove the repetitive bounding boxes around each object and only keep the box with the highest score.

Since the development of YOLO at 2015, there have been several versions and varieties of this model proposed by introducing different improvements and alternations to the original model. For instance, analyzing the error of the original YOLO approach showed a number of flaws including the production of a large number of localization errors and having a lower recall rate in comparison with the region proposal-based methods [22]. In YOLOv2 [22] several techniques are employed in order to improve upon the original version. These techniques include high-resolution classifier, batch normalization, direct location prediction, dimension clusters, and multi-scale training. The Darknet-19 network was used as the classification backbone which consists of five max-pooling layers and 19 convolutional layers.

Later, YOLOv3 [23] was introduced which further improved the robustness and efficiency of the previous versions. In this version, the softmax layers are replaced with independent logistic classifiers and the binary cross-entropy loss is utilized in the classification process. The Darknet-19 model is replaced with Darknet-53 and detections are performed in three different scales in order to deal with small objects, which was a problem for YOLOv2. As opposed to the YOLOv2, which used the softmax function, YOLOv3 uses a multi-label classification approach and each bounding box can belong to several classes at the same time.

The latest official version of this method is YOLOv4 which is improves the performance of the previous version both in terms of mean average precision

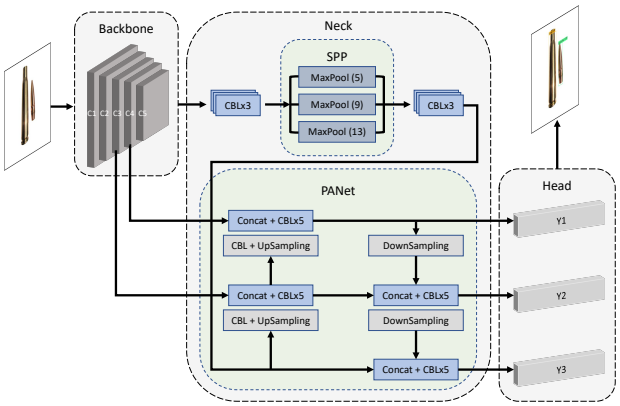


Fig. 2. The system architecture of the YOLOv4 model.

(mAP) and speed. As seen in the fig. 1 the architecture of YOLOv4 consists of three distinct components, namely, the backbone, the neck, and the head. The backbone network for feature extraction is the CSPDarknet53 which is used for splitting the current layer into two parts. The first part passes through the convolution layers while the second part doesn't and the results are aggregated at the end. The neck is the intermediate section between the backbone and the head and contains modified versions of the path aggregation network (PANet) and spatial attention module with the purpose of having a higher accuracy by information aggregation. The head of the architecture represents the dense prediction block, which is used to locate the bounding boxes and final classification. Similar to YOLOv3 the bounding box locations and object probabilities are calculated as the output of the model.

Several additional sets of techniques are applied in YOLOv4 in order to further enhance the detection results, which are called bag of freebies and bag of specials. The bag of freebies consists of various approaches, such as cut mix and mosaic data augmentation, drop block for regularization, self-adversarial training, and random training shapes. On the other hand, the so-called bag of specials is a set of post-processing modules designed to considerably improve the accuracy with a slight increase in the inference time. This set of techniques includes different modules including mish activation, cross-stage partial connections (CSP), the spatial pyramid pooling (SPP) block, the spatial attention module (SAM), path aggregation network (PANet), and the distance IoU non-maximum suppression. Figure 2 illustrates the detailed architecture of YOLOv4 object detection method.

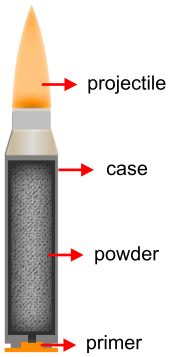


Fig. 3. Basic Components of Ammunition.

3 Ammunition Component Detection and Classification Using Deep Learning

We apply one of the most representative deep learning methods to date, the YOLOv4, for ammunition component classification in visual and x-ray ammo images. The training data sets are manually created using publicly available images. Specifically, two training data sets are created corresponding to the training samples from visual and x-ray images mostly captured from 50 calibers by vision cameras, transmission x-rays, or back-scatter x-rays. Each image contains one or more ammo components or full-ammo. The goal is to detect each full-ammo or ammo component and classify it into one of the two classes, namely Material Documented as Safe (MDAS) and Material Potentially Possessing Energetic Hazard (MPPEH). These classes indicate whether the scrap piece is safe or potentially hazardous.

The basic components of ammunition are the case, gunpowder, primer, and projectile. The case is a usually cylindrical container that holds the ammunition components together as one piece. Various materials are used for the case, such as steel, copper, brass, plastic, and paper. The powder is a chemical mixture and is converted to an expanding gas when ignited. This component of ammunition is considered to be explosive hazard and should not remain in the scrap during the recycling process. The primer is an explosive chemical compound that is used to ignite gunpowder when it is struck by a firing pin. The projectile is the part of ammunition that is expelled from the barrel which is usually referred to as a bullet. Figure 3 demonstrates the main components of ammunition.

Typically, ammunition scrap pieces are loose casings, loose projectiles, or full ammo with case and projectile in place, but missing the primer. Blown out casings and burst projectiles are also likely to appear among the metallic scrap. In an ideal scenario, there is no energetics left in the ammunition scrap and only the metal parts go through the recycling process. However, in real-world situations, some of the ammunition is not destroyed well enough during the incineration and the powder remains inside the case. This is considered to be

potentially dangerous as there is an explosive hazard among the metallic scrap. Therefore, there is a need for inspecting the scrap in order to make sure all the scrap pieces are safe to go through the recycling process. Here, we have constructed a dataset of visual images and a dataset of x-ray images with the purpose of training a deep learning model for the detection and classification of safe and unsafe scrap pieces.

3.1 Ammunition Component Detection in Visual Images

We have gathered and annotated a sufficient number of visual images of ammunition components and full-ammo to be used as training and testing samples. The YOLOv4 object detection method is applied in order to train three classes of scrap pieces. Specifically, full-ammo, projectile, and case are the three classes that are used to train the deep CNN. Since there is a good chance of a full-ammo to contain energetics in its case, we have considered this class to represent the unsafe samples, called Material Potentially Possessing Energetic Hazard (MPPEH). On the other hand, the separated casings or projectiles are assumed to be safe with a high degree of certainty and are classified as Material Documented as Safe (MDAS).

In addition to the original YOLOv4 structure, we have also used a shallower lightweight network, called tiny-YOLOv4 as the backbone in order to increase the computational speed with a negligible drop in the accuracy [17]. The lightweight structure contains 29 layers compared to the original one with more than a hundred. The Cross Stage Partial (CSP) model is derived from the DenseNet architecture, which concatenates the previous input with the current input prior to reaching to the dense layer. The backbone of the tiny-YOLOv4 includes an input layer followed by 18 convolutional layers, 9 routes, 3 max-pooling layers, and a detection layer based on YOLOv3 at the end. Several features of each input image are extracted by the convolutional layers. There are interchangeable 3×3 and 1×1 receptive filters striding over the input image to generate feature maps, which are passed through other layers of the network. The leaky-ReLU activation function is applied at the convolutional layers in order to increase the feature size. Routes are designed to improve the gradient flow throughout the layers.

3.2 Ammunition Component Detection in X-Ray Images

Visual images represent human vision and are not capable of representing some of the most important aspects of a visual scene. In the case of ammunition scrap, the visual cameras are not able to capture any information from the inside of the ammunition casings. However, there might be energetics still remaining inside of the scrap pieces even if they are separated. Therefore, other visual modalities such as x-ray penetration can help increase the accuracy and reliability of the inspection. The x-ray images clearly capture the gunpowder inside the scrap pieces, which makes them a beneficial resource in the classification of safe/unsafe samples.

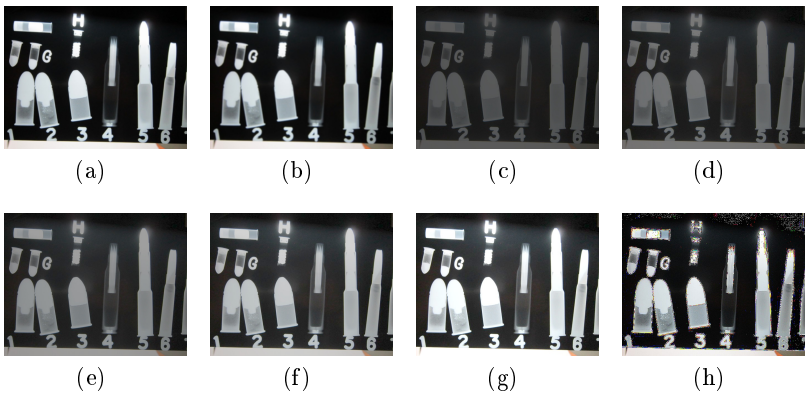


Fig. 4. The sample images generated using data augmentation. (a) and (b) are the results of averaging with kernel sizes 3 and 5, respectively. (c), (d), (e), (f), (g) are the results of power law transformation with gamma values 0.40, 0.45, 0.5, 0.55, and 0.6, respectively, and (h) is the result of sharpening.

We have gathered a number of x-ray images of ammunition to form a dataset for training the deep CNN model. Methods based on deep learning require a large number of training data in order to tune the parameters and learn the features. Since the number of acquired x-ray images is not sufficient to train a deep learning model we have applied a number of data augmentation techniques in order to increase the number of training samples and highlight the features of interest. We have applied a number of augmentation techniques in the form of spatial transformations, such as histogram equalization, power law, averaging, sharpening, negative, and Gaussian blurring on the input x-ray images. Image sharpening is applied as the addition of the original image and the high frequency for the purpose of enhancing the edges in images with poor qualities. The gamma power of image intensities is calculated to compute the power law transformation, which is applied to manipulate the image contrast and perform calibration. Figure 4 illustrates a few examples of data augmentation techniques applied on an x-ray image.

4 Experiments

We conduct a number of experiments using two image datasets to evaluate the performance of YOLOv4 deep learning method in ammunition component detection and classification. The two datasets are created by collecting visual and x-ray images of ammunition components in addition to full-ammo. Specifically, we call the first dataset **Visual Ammunition Component Detection** (VACD), which contains 162 visual images for training along with 50 images for testing. Each image contains somewhat between one and thirteen scrap pieces where

each piece is either a full-ammo or an ammunition component such as casing or projectile.

The second dataset is named **X-Ray Ammunition Component Detection** (XACD), which is a collection of 108 x-ray images obtained by applying data augmentation techniques on the initial 12 images. From the 108 x-ray images, 72 are used as training samples and the remaining 36 are utilized as test data. For each visual or x-ray image, the ground-truth bounding boxes are manually annotated and labeled. Table 1 summarizes the three training data sets.

Table 1. The Ammunition Component Datasets

Dataset	Imaging modality	Training samples	Testing samples
VACD	Visible	162	50
XACD	X-Ray	72	36

For evaluating the quantitative results of the object detection method, three performance measures are utilized as follows:

$$\begin{cases} PRE = T_P/(T_P + F_P) \\ REC = T_P/(T_P + F_N) \\ F_1 = 2 \times (PRE \times REC)/(PRE + REC) \end{cases} \tag{1}$$

where T_P, F_P, and F_N are the true positive, false positive, and false negative instances, respectively. PRE, REC, and F_1 refer to precision, recall, and F1-score, respectively.

We used a public github repository [1] for the experiments. For parameter settings, the batch sizeis 64, learning rate is 0.001, momentum is 0.973, and decay is selected to be 0.0005. The tiny-YOLOv4 backbone is a CSP that contains CBL, cross-stage, and residual features along with skip connection layers. It applies two detectors at the end head. More details can be found at the repository [1]. The first experiments were carried out using the VACD dataset for training a tiny-YOLOv4 model. Table 2 shows the results of the experiments conducted on the VACD data in terms of the performance measures. The confidence threshold is set to 0.25 and the intersection over union (IoU) threshold is computed as 54.63%. The mean average precision for this dataset reached 84.94% and the total detection time was 6 seconds. Figure 5 shows a number of detection results of testing the tiny-YOLOv4 model against the images that are unseen during the training.

The second set of experiments was carried out using the XACD dataset for training a tiny-YOLOv4 model. Table 3 shows the results of the experiments conducted on the XACD data in terms of the performance measures. The confidence threshold is set to 0.25 and the intersection over union (IoU) threshold is computed as 50%. The mean average precision for this dataset reached 99.92% and the total detection time was 8 seconds. Figure 6 illustrates sample detection

Table 2. The quantitative results of testing tiny-YOLOv4 on the VACD data

	TP	FP	FN	Precision	Recall	F1-score
Full-ammo	159	50	21	76.08%	88.33%	81.75%
Casing	92	104	10	46.94%	90.2%	61.74%
Projectile	84	25	9	77.06%	90.32%	83.17%
Total	335	179	40	65%	89%	75%

Fig. 5. Sample detection and classification results of applying tiny-YOLOv4 on the test images of the VACD dataset.

results of testing the tiny-YOLOv4 model against the images that are unseen during the training.

Table 3. The quantitative results of testing tiny-YOLOv4 on the XACD data

	TP	FP	FN	Precision	Recall	F1-score
Full-ammo	499	21	1	95.96%	99.8%	97.84%
Casing	41	6	7	87.23%	85.42%	86.32%
Projectile	18	0	4	100%	81.82%	90%
Total	558	27	12	95%	98%	97%

The hardware specification used for the experiments is a 3.4 GHz processor, 16 GB RAM, and an Nvidia GTX-745 graphics processing unit (GPU). The time spent on the training of the tiny-YOLOv4 model using the VACD dataset for 6000 iterations was approximately four hours. Figure 7 illustrates the growth of

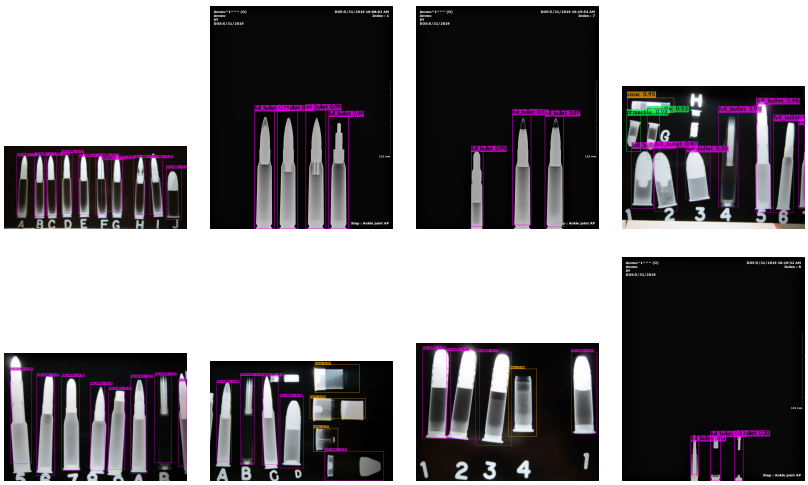


Fig. 6. Sample detection and classification results of applying tiny-YOLOv4 on the test images of the XACD dataset.

mean average precision (mAP) in terms of training iterations using the VACD data.

Our findings in this study demonstrate the need of deep learning models for a considerable amount of data and computational resources in comparison with the traditional methods. Additionally, however necessary for supervised learning, the manual labeling and annotating the collected data is a tedious and time-consuming process. The extracted features are not easily interpretable by human experts and the complexity of numerous hidden layers in the deep network architectures increases as the model uses more parameters. Another drawback of the deep learning methods is their low ability of generalization.

Despite the limitations, when tested on data similar to the training samples, a well-trained deep model demonstrates high performance in the task that it is trained for. All the YOLOv4 models trained using the collected datasets showed great accuracy in detecting ammunition components and full-ammo when tested against visible or x-ray images. The YOLOv4 method proved to be fast enough for use in applications with real-time requirements. Visual and x-ray imaging systems can be deployed on the inspection site. The trained models can be applied in real-time during the ammunition metallic scrap inspection in order to discriminate the Material Documented as Safe (MDAS), which involves the projectile and casing samples, and Material Potentially Possessing Energetic Hazard (MPPEH) which refers to the full-ammo class. The potentially dangerous scrap pieces should be separated for further inspections.

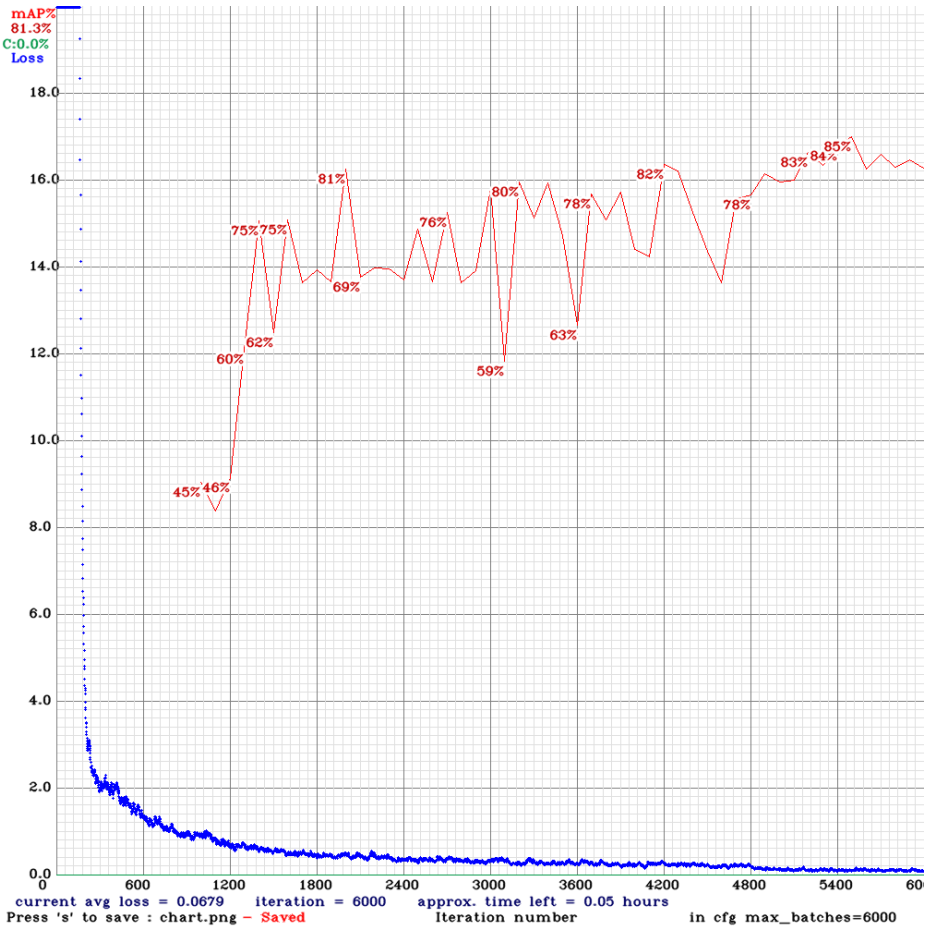


Fig. 7. changes in the loss and mAP in 6000 iterations of training tiny-YOLOv4 against the VACD data.

5 Conclusions

The representative YOLOv4 object detection method is applied for the task of ammunition component detection and classification in images of the visible and the x-ray range. The purpose of this classification is to aid with the automatic inspection of ammunition scrap in the process of reducing dangerous properties prior to recycling. Two image databases are gathered and annotated for the task of object detection. First dataset, Visual Ammunition Component Detection (VACD), is a set of visible images of ammunition components and full-ammo pieces. The second database, X-Ray Ammunition Component Detection (XACD), is a set of x-ray images of ammunition components, which is enhanced by several data augmentation techniques. The x-ray images are able to illustrate

the insides of the ammunition casings, which helps indicate whether the scrap piece contains any energetics or not. As a general rule of thumb, we consider full-ammo to be explosive hazard due to the possibility of remaining energetics inside. The potentially dangerous scrap pieces should be separated and go through the incineration process again before being moved to the recycling unit. The experimental evaluations using the collected datasets demonstrate the feasibility of the YOLOv4 method in object detection and classification in real-time applications.

References

1. Alexey, Redmon, J., Sinigardi, S., cyy, Hager, T., JaledMC, Maaz, M., Zhang, V., Alasuutari, J., Kahn, P., IlyaOvodov, Veitch-Michaelis, J., Dujardin, A., Aughey, J., Patel, A., duohappy, Aven, Smith, D., White, J., MarvinKlemp, Özipek, E., Babaei, A., Gąsiorzewski, B., Dev-Nash, Double, HagegeR, Daras, G., Popovych, I., Pizzorno, J.A., Giordano, M.: Alexeyab/darknet: Yolov4 (Oct 2021). https://doi.org/10.5281/zenodo.5622675, `https://doi.org/10.5281/zenodo.5622675`
2. Bochkovskiy, A., Wang, C.Y., Liao, H.Y.M.: Yolov4: Optimal speed and accuracy of object detection. arXiv preprint arXiv:2004.10934 (2020)
3. Chen, S., Liu, C.: Clustering-based discriminant analysis for eye detection. IEEE Transactions on Image Processing **23**(4), 1629–1638 (2014)
4. Chen, S., Liu, C.: Eye detection using discriminatory haar features and a new efficient svm. Image and Vision Computing **33**, 68–77 (2015)
5. Faruque, M.O., Ghahremannezhad, H., Liu, C.: Vehicle classification in video using deep learning. In: Machine Learning and Data Mining in Pattern Recognition, MLDM. pp. 117–131. ibai publishing, Leipzig (2019)
6. Feng, D., Harakeh, A., Waslander, S.L., Dietmayer, K.: A review and comparative study on probabilistic object detection in autonomous driving. IEEE Transactions on Intelligent Transportation Systems (2021)
7. Feraco, S., Bonfitto, A., Amati, N., Tonoli, A.: Redundant multi-object detection for autonomous vehicles in structured environments. Communications-Scientific letters of the University of Zilina **24**(1), C1–C17 (2022)
8. Ghahremannezhad, H., Shi, H., Liu, C.: Robust road region extraction in video under various illumination and weather conditions. In: 2020 IEEE 4th International Conference on Image Processing, Applications and Systems (IPAS). pp. 186–191. IEEE (2020)
9. Ghahremannezhad, H., Shi, H., Liu, C.: Automatic road detection in traffic videos. In: 2020 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking (ISPA/BDCloud/SocialCom/SustainCom). pp. 777–784. IEEE (2020)
10. Ghahremannezhad, H., Shi, H., Liu, C.: A new adaptive bidirectional region-of-interest detection method for intelligent traffic video analysis. In: 2020 IEEE Third International Conference on Artificial Intelligence and Knowledge Engineering (AIKE). pp. 17–24. IEEE (2020)
11. Ghahremannezhad, H., Shi, H., Liu, C.: A real time accident detection framework for traffic video analysis. In: Machine Learning and Data Mining in Pattern Recognition, MLDM. pp. 77–92. ibai publishing, Leipzig (2020)

12. Ghahremannezhad, H., Shi, H., Liu, C.: A new online approach for moving cast shadow suppression in traffic videos. In: 2021 IEEE International Intelligent Transportation Systems Conference (ITSC). pp. 3034–3039. IEEE (2021)
13. Guo, J., Chen, P., Jiang, Y., Yokoi, H., Togo, S.: Real-time object detection with deep learning for robot vision on mixed reality device. In: 2021 IEEE 3rd Global Conference on Life Sciences and Technologies (LifeTech). pp. 82–83. IEEE (2021)
14. Liu, C.: A bayesian discriminating features method for face detection. IEEE transactions on pattern analysis and machine intelligence **25**(6), 725–740 (2003)
15. Liu, G., Shi, H., Kiani, A., Khreishah, A., Lee, J., Ansari, N., Liu, C., Yousef, M.M.: Smart traffic monitoring system using computer vision and edge computing. IEEE Transactions on Intelligent Transportation Systems (2021)
16. Liu, L., Ouyang, W., Wang, X., Fieguth, P., Chen, J., Liu, X., Pietikäinen, M.: Deep learning for generic object detection: A survey. International journal of computer vision **128**(2), 261–318 (2020)
17. Montalbo, F.J.P.: A computer-aided diagnosis of brain tumors using a fine-tuned yolo-based model with transfer learning. KSII Transactions on Internet & Information Systems **14**(12) (2020)
18. Niranjan, D., VinayKarthik, B., et al.: Deep learning based object detection model for autonomous driving research using carla simulator. In: 2021 2nd International Conference on Smart Electronics and Communication (ICOSEC). pp. 1251–1258. IEEE (2021)
19. O'Keeffe, S., Villing, R.: Evaluating pruned object detection networks for real-time robot vision. In: 2018 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC). pp. 91–96. IEEE (2018)
20. Osuna, E., Freund, R., Girosit, F.: Training support vector machines: an application to face detection. In: Computer vision and pattern recognition, 1997. Proceedings., 1997 IEEE computer society conference on. pp. 130–136. IEEE (1997)
21. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 779–788 (2016)
22. Redmon, J., Farhadi, A.: Yolo9000: better, faster, stronger. arXiv preprint (2017)
23. Redmon, J., Farhadi, A.: Yolov3: An incremental improvement. arXiv preprint arXiv:1804.02767 (2018)
24. Shi, H., Ghahremannezhad, H., Liu, C.: Anomalous driving detection for traffic surveillance video analysis. In: 2021 IEEE International Conference on Imaging Systems and Techniques (IST). pp. 1–6. IEEE (2021)
25. Shi, H., Ghahremannezhadand, H., Liu, C.: A statistical modeling method for road recognition in traffic video analytics. In: 2020 11th IEEE International Conference on Cognitive Infocommunications (CogInfoCom). pp. 000097–000102. IEEE (2020)
26. Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features. In: Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on. vol. 1, pp. I–I. IEEE (2001)
27. Zou, Z., Shi, Z., Guo, Y., Ye, J.: Object detection in 20 years: A survey. arXiv preprint arXiv:1905.05055 (2019)

Feature Extractor Enhancement by Structural Modifications

Sijin Ren[1,2][0000−0002−7853−8674], Cheryl Li[2][0000−0002−2228−6834], and Xiao Mei[3][0000−0002−1035−6663]

[1] Yale University, New Haven CT 06510, USA
sijin.ren@yale.edu
[2] University of New Haven, New Haven CT 06516, USA
cli@newhaven.edu
[3] Shanghai Maritime University, Shanghai 201306, China
xiaomei@shmtu.edu.cn

Abstract. Transfer learning networks have gained massive popularity recently. The mainstream transfer learning architecture consists of two parts: the first part is a feature extractor to extract features from the input data, and the second part is to process those extracted features. The feature extractor plays a decisive role in the performance of the transfer learning network. Despite its importance, the study about the improvement of feature extractor for transfer learning networks is lacking. Here we scrutinized the enhancement of the feature extractor via structural modifications. To this end, we have taken three structural modification methods into consideration. They were simplifying the feature extractor by deleting its last few layers, fine-tuning its first few layers, and fine-tuning its last few layers, respectively. The results showed that structural modifications can improve the feature extractor capability and enhance the transfer learning network performance subsequently. Our specific examples showed that the image classification accuracy of a well-performed transfer learning network was improved by 2.6% maximally. Without any additional computational cost, the accuracy was increased by 0.77%. We conclude that the performance of feature extractor in transfer learning network can be improved via structural modifications.

Keywords: Transfer learning · Feature extractor · Structural modification · Machine learning

1 Introduction

In recent years, significant progress has been made in deep learning tasks due to the availability of the state-of-art convolutional neural networks (CNNs) [1–3]. In addition, increasingly powerful computational resources have led to the widespread use of CNNs in various applications. The performances of CNNs are highly correlated with the amount of training data. Those widely adopted CNNs were normally trained using large datasets, e.g., the ImageNet dataset [4].However, in real-world applications, the volume of training data is often limited [5].

Thus, the performances of CNNs decline markedly in some practical applications where training data is limited.

In contrast, transfer learning networks are robust against the shortage of training data [6]. The transfer learning approach reuses a source network that was already well-trained using enough images (i.e., the source dataset). In the transfer learning process, the first n layers of the well-trained source network are frozen and transferred to an m-layer target network (m > n). Then, the remaining (m-n) layers in the target network are trained by a relatively small target dataset. The transferred part functions as a strong feature extractor [7, 8]. The feature extractor's ability is directly transferred from the source network to the target network without further training. Therefore, a transfer learning network can obtain a good performance without the availability of a large amount of training data. With this special feature, transfer learning has gained popularity in various fields, and many studies have demonstrated its extremely encouraging performance with limited training data [5, 9, 10].

The feature extractor draws features from the input data, and the training of the rest of the transfer learning network is based on the features extracted. An effective feature extractor is the key component in a transfer learning network. A poor feature extractor will lead to a drastic network performance deterioration [7]. This is referred as "negative transfer"in literature [7]. Despite the critical importance of the feature extractor, there is a lack of research in the improvement of feature extractor performance. Previous studies have shown that modifi cations of network structure can significantly enhance the CNN performances [11, 12, 3, 1]. Inspired by these studies, in this paper we examined how structure modification of a feature extractor could improve its performance.

Yosinski et al.the generality versus specificity of the feature extractor of a CNN. They found that transferability is negatively impacted by the specialization of higher layers to their original task at the expense of performance on the target task [8]. In a radar target detection study, Bralich et al. also found that the transfer learning network performance would deteriorate if the features were extracted from the deeper layers (e.g., four or more layers) [13]. Inspired by these research results, this study examined how simplification of the structure of a feature extractor can adversely improve the learning performance of a transfer learning network. We removed the last part of the feature extractor to avoid the over extraction of the inappropriate features.

In transfer learning applications, fine-tuning is also widely used to enhance the performance of a feature extractor [14, 15]. This method frees the parameters in the feature extractor and turns them from frozen to trainable. During the training process, the feature extractor takes the frozen parameters as the initial values. Then, the parameters are trained jointly with other raw parameters using the target data. As a result, fine-tuned layers are shaped by the target dataset, and the features of the target domain are extracted. Since the last few layers of the feature extractor are more focused on the source domain than the front layers, fine- tuning the later layers is sufficient [16, 17, 15]. Although this strategy is widely used, how exactly the fine-tuning of these layers will affect the learn-

ing performance has not been well studied. The second structure modification addressed in this paper aims to answer this question.

Filters from the first layers of a feature extractor are like Gabor filters or color blobs that focus on extracting general features. Thus, the first layers of a feature extractor can be safely applied across domains [8]. However, Aljundi and Tuytelaars pointed out that domain shift effects occur at the very first layer in an unsupervised CNN [18]. Shirokikh et al. also found that fine-tuning the first few layers in a U-net transfer learning network significantly outperformed the fine-tuning of its last layers in a recent MRI segmentation study [19]. They indicated that the first layers contain more source domain-oriented information than the last layers, once a U-net is well-trained. For transfer learning networks used for image classification tasks or similar super-vised network structures, to our best knowledge, no study has investigated the specificity/generalization of the first layers in the feature extractor. Thus fine-tuning the first few layers of the feature extractor is the third modification to be considered in this study.

The effectiveness of these three structural modifications to the feature extractor was examined in this paper. The performance of the feature extractor was measured according to the image classification accuracy of a transfer learning network with the feature extractor. The results obtained demonstrated that all three modifications can improve network performance. The simplification strategy enhanced the feature extractor slightly as well as reducing the computational expense. Fine-tuning the last layer of the feature extractor improved the feature extractor maximally with more than two million fine-tuned parameters. Fine-tuning the first few layers also exhibited an improved performance, in addition to the significant reduction of the number of tuned parameters. Hence, we conclude that the proposed structural modification methods can make the feature extractor learn better and enhance the network performance.

2 Datasets

Both the training and testing images were drawn from an open-source dataset containing classified images [20] . Each category involved more than 2000 training images and approximately 500 testing images. In the training dataset, there were 4000 images randomly and evenly selected from four specific classes, i.e., the “buildings,” “forest,” “glacier,” and “sea” categories. The testing dataset contained 1,917 images from the selected four classes as well. There was no overlap between the training and testing images or different types of images, and all images are 100 × 100-pixel RGB images. Sample images from the four classes are shown in Fig 1.

3 Proposed Transfer Learning Network

3.1 Source Network and Feature Extractor

In this study, VGG16 was employed as the source network, and the proposed feature extractor was a part of VGG16. VGG16 is one of the most popular source

Fig. 1. Sample images from four classes

networks in transfer learning research community [5, 16, 21] and can generalize well to extensive datasets [3].

VGG16 was well-trained using a subset of ImageNet. This subset contained 1.2 million RGB images from 1000 categories. In VGG16, the image was passed through a bunch of convolutional layers with 3×3 convolutional filters, which was the smallest filter able to capture the notion of relative position. The convolution stride was set to 1 pixel. The spatial pooling was executed by five max-pooling layers. The max-pooling filters were in a 2×2-pixel window with a stride of 2. Two fully-connected layers were following the last pooling layer, and each fully-connected layer involved 4096 neurons. The SoftMax classifier, i.e., the last layer, contained 1000 channels because VGG16 was designed to classify 1000-class images.

In VGG16, over 119 million parameters were in the last three layers, representing approximately 88% of all parameters. To avoid an excessive number of parameters, the feature extractor employed in this study was the VGG16 network without the last three layers. As shown in Fig 2, the target feature extractor applied was the first 18 layers of VGG16, i.e., a combination of all convolutional and max-pooling layers.

3.2 Architecture of Proposed Transfer Learning Network

The evaluation of the feature extractor modifications was based on the performance of the proposed transfer learning network. The proposed network architecture is shown in Fig 2. The first part of the proposed network was the feature extractor. The feature extractor was followed by a global average pooling (GAP) layer, one fully-connected layer, and one SoftMax classifier. These three layers and the feature extractor comprised the proposed transfer learning network. As shown in Fig. 2, the layers in the solid box replace the layers in the dashed box.

The GAP layer was employed to flatten the tensor from the feature extractor, where each channel of the feature tensor was averaged to a value. Thus, the tensor size was reduced considerably and overfitting was constrained. A smaller 512-neuron fully-connected layer came after the GAP layer. The final layer was the SoftMax classifier layer whose output is a $1 \times 1 \times 4$ vector, where each element represented a single image category.

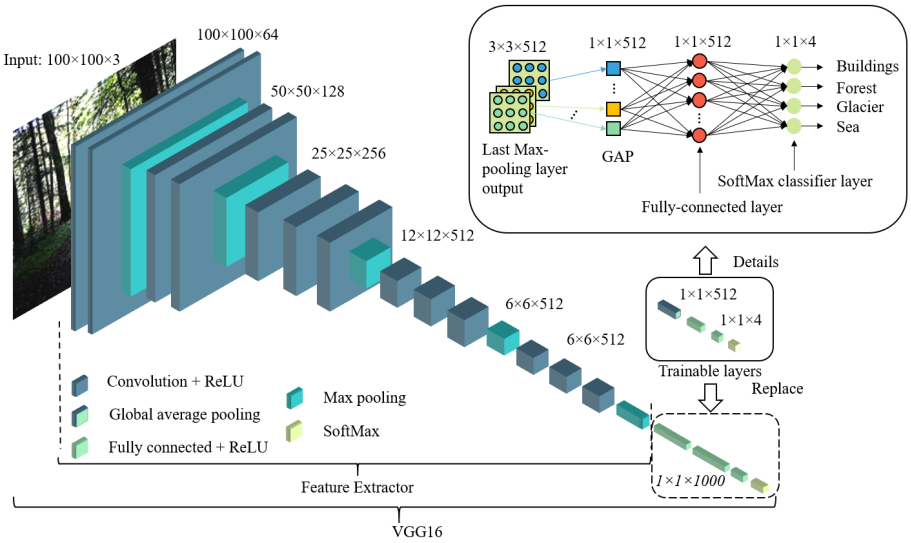


Fig. 2. VGG16, feature extractor, and proposed transfer learning network

Table 1. Structure and parameters distribution of the proposed transfer learning network

Blocks	Network layers	Output dimension	Parameter number in the layer
	Input layer	100×100×3	0
1	Convolutional layer 1	100×100×64	3×9×64+64=1792
1	Convolutional layer 2	100×100×64	64×9×64+64=36928
1	Pooling layer (Max-pooling)	50×50×64	0
2	Convolutional layer 3	50×50×128	64×9×128+128=73856
2	Convolutional layer 4	50×50×128	128×9×128+128=147584
2	Pooling layer (Max-pooling)	25×25×128	0
3	Convolutional layer 5	25×25×256	128×9×256+256=295168
3	Convolutional layer 6	25×25×256	256×9×256+256=590080
3	Convolutional layer 7	25×25×256	256×9×256+256=590080
3	Pooling layer (Max-pooling)	12×12×256	0
4	Convolutional layer 8	12×12×512	256×9×512+512=1180160
4	Convolutional layer 9	12×12×512	512×9×512+512=2359808
4	Convolutional layer 10	12×12×512	512×9×512+512=2359808
4	Pooling layer (Max-pooling)	6×6×512	0
5	Convolutional layer 11	6×6×512	512×9×512+512=2359808
5	Convolutional layer 12	6×6×512	512×9×512+512=2359808
5	Convolutional layer 13	6×6×512	512×9×512+512=2359808
5	Pooling layer (Max-pooling)	3×3×512	0
Trainable	Global average pooling layer	512	0
Trainable	Fully connected layer	512	512×512+512=262656
Trainable	SoftMax classifier	4	512×4+4=2052

Before training, the parameters in the fully-connected layers and SoftMax classifier were initialized using the default kernel initializer [22], and the default bias was set to be 0. Other parameters in the feature extractor were frozen and untrainable. The parameter distribution in the proposed network was shown in Table 1. The trainable parameters counted less than 2% of all parameters. From this table, the feature extractor was divided into five blocks, where two or three jointed convolutional layers and the following max-pooling layer comprised a single block.

3.3 Training Settings

The goal of the training process was to reduce the loss function as much as possible. The sparse categorical cross-entropy loss function has been highly recommended for classification tasks when data in different categories are mutually exclusive [23]. Thus, the sparse categorical cross-entropy loss function was employed in this study. The RMSprop optimizer, demonstrating fast convergence [24], was implemented.

In training, the learning rate was always set to 0.001, the default value of the RMSprop optimizer. The number of epochs in all training cases was set to 20. In experiments, we found 20 epochs can lead to a well-trained transfer learning network without requiring excessive training time.

4 Feature Extractor Modifications

In this section, the structural modifications applied to the feature extractor are described in detail.

4.1 Simplifying the Feature Extractor

In the simplifying modification, the last few layers of the feature extractor were deleted to eliminate the undesired features. The goal was to improve the feature extractor's generalizability without sacrificing its performance. During the simplifying process, the last convolution layer was deleted first. Then, the last six convolution layers were removed one by one. In this process, other parts of the proposed network remained the same. Fig. 3 shows the details of the simplified feature extractor with the last convolutional layer removed.

4.2 Fine-tuning the Last layers of Feature Extractor

Fine-tuning the last convolution layers in the feature extractor was considered in this subsection. In this approach, the parameters in the last convolutional layers of the feature extractor were converted from frozen to trainable, and these parameters were initialized by the values inherited from VGG16. Other trainable parameters were initialized randomly by the default glorot_uniform kernel initializer with a bias of zero. Both kinds of parameters were trained together using

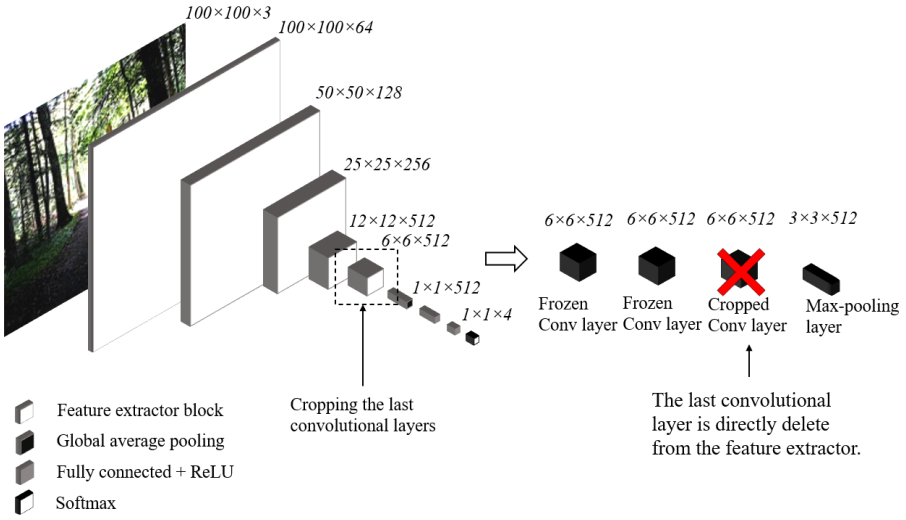


Fig. 3. Simplified the feature extractor with the last convolutional layer removed

the target training data. The modified feature extractor with the fine-tuned final layer were shown in Fig 4. Table 1 shows that the last convolutional layers from the feature extractor contain over two million parameters each. Although fine-tuning more layers would be more beneficial, handling such a large number of parameters required a massive amount of training data. The available training data was limited. No more than two layers were fine-tuned to avoid overfitting[14].

4.3 Fine-tuning First layers of Feature Extractor

Fine-tuning the first few layers is presented in this subsection. Fig 4 shows the proposed network and feature extractor with the fine-tuned first layer. Compared to fine-tuning the final part of the feature extractor, fine-tuning the first part can dramatically reduce the number of parameters to be fine-tuned. Shirokikh et al. pointed out that U-net performance improvement was notable when the first two layers of the network are fine-tuned [19]; thus, we considered fine-tuning the first one or two layers.

5 Results

This section describes the impact of the three structural modifications on the performance of the feature extractor. These three modifications were mutually exclusive to each other. The feature extractor performance was represented by the classification accuracy obtained on the testing dataset. To ensure the reliability of the results, the experiment was repeated five times for each case. Then, the

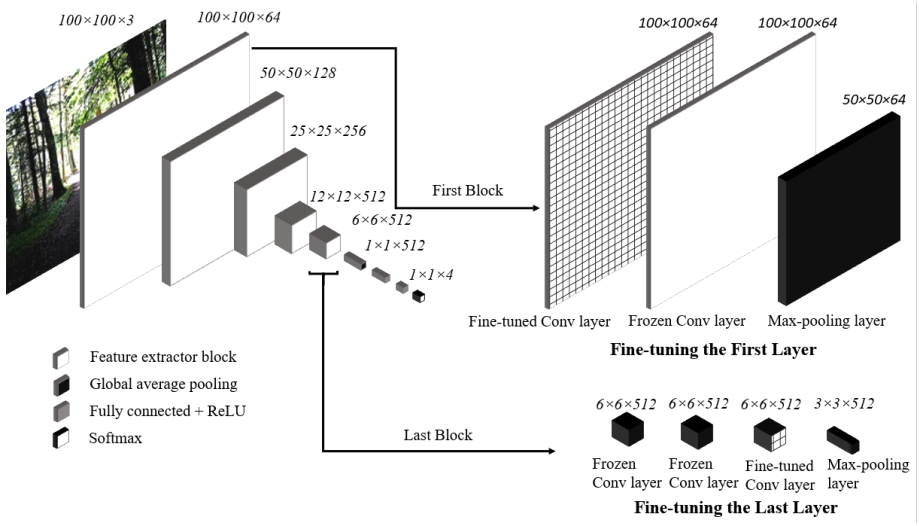


Fig. 4. Proposed network with fine-tuned the first and the last convolutional layer in the feature extractor

average accuracy was taken as the final classification accuracy. In addition, the accuracy of the proposed network was compared to the classification accuracy obtained by a network with an unmodified feature extractor. To visualize the impact of features by the considered modifications, some features from different parts of the extractor are also illustrated.

5.1 Results on Simplified Feature Extractor

The classification accuracies varied as different layers were removed from the feature extractor as Table 2 shows. When the final convolutional layer was cropped, classification accuracy increased from 94.23% to 94.96%, a relatively 0.77% enhancement. It is the best network performance was obtained by simplifying the feature extractor. Cropping the last block (three convolutional layers) was also beneficial, i.e., a small enhancement to classification accuracy was obtained (94.23% to 94.7%). However, when two convolutional layers were cropped, network performance was reduced slightly. Cropping more than three layers negatively affected the network. In the worst-case scenario, classification accuracy was reduced from 94.23% to 89.21% when the last five convolutional layers were removed.

5.2 Results on Fine-tuning the Last Part of the Feature Extractor

The performance of the proposed transfer learning network with the last few layers fine-tuned in the feature extractor is shown in Table 3. Fine-tuning the

Table 2. Image classification accuracy vs. number of removed layers from the feature extractor

Deleted layers	Accuracy	Relative change
No modification	0.9423	0
Crop 1 conv layer	0.9496	**+0.77%**
Crop 2 conv layers	0.9370	-0.56%
Crop 3 conv layers (crop 1 block)	0.9470	+0.50%
Crop 4 conv layers	0.9281	-1.51%
Crop 5 conv layers	0.8921	-5.32%
Crop 6 conv layers (crop 2 blocks)	0.9189	-2.48%

last convolutional layer in the feature extractor obtained the best network performance. Here, classification accuracy increased from 94.23% to 96.68%, a relatively 2.6% improvement. To demonstrate the impact of the modifications on extracted features, a comparison of the features from the last layer before and after fine-tuning are shown in Fig 5. As shown in Fig 5, the features were transformed significantly. Most of the non-zero pixel values were changed to zeros, and only a single non-zero value was left. However, when an additional layer was fine-tuned, a dramatic drop in accuracy was observed. The classification accuracy was reduced by 23.39%.

Table 3. Classification accuracy vs. fine-tuning last layers of feature extractor

Deleted layers	Accuracy	Relative change
No modification	0.9423	0
Fine-tuning the last layer	0.9668	**2.60%**
Fine-tuning the last two layers	0.7219	-23.39%

5.3 Results on Fine-tuning the first layers of the Feature Extractor

The accuracy of the proposed network varied when the feature extractor's first layers were fine-tuned. When the first layer was fine-tuned, image classification accuracy was improved from 94.23% to 95.28%. To visualize the adjustments of the extracted features, a comparison of the features from the first layer before and after fine-tuning is shown in Fig.6. In terms of features, the original outputs of the first layer were primarily focused on the building's outline and grayscale. Some features were overlapped. After fine-tuning, the output of the first layer was more diverse, and no feature overlapped others. By fine-tuning an additional layer, network performance was optimized and the classification accuracy increased from 94.23% to 94.53%. The effects of optimization were not as obvious as that of fine-tuning only the first layer.

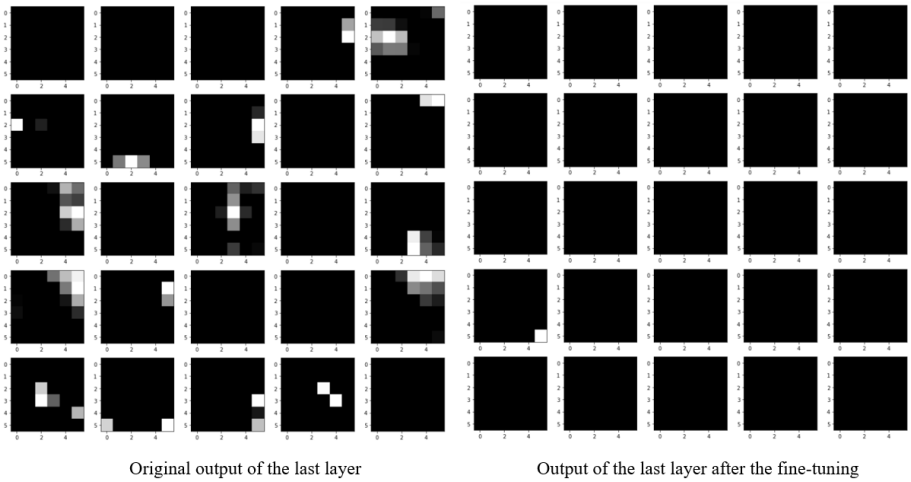


Fig. 5. Comparison of features from the last convolutional layer

Table 4. Classification accuracy vs. fine-tuning first layers of feature extractor

Deleted layers	Accuracy	Relative change
No modification	0.9423	0
Fine-tuning the first layer	0.9528	**1.11%**
Fine-tuning the first two layers	0.9453	0.32%

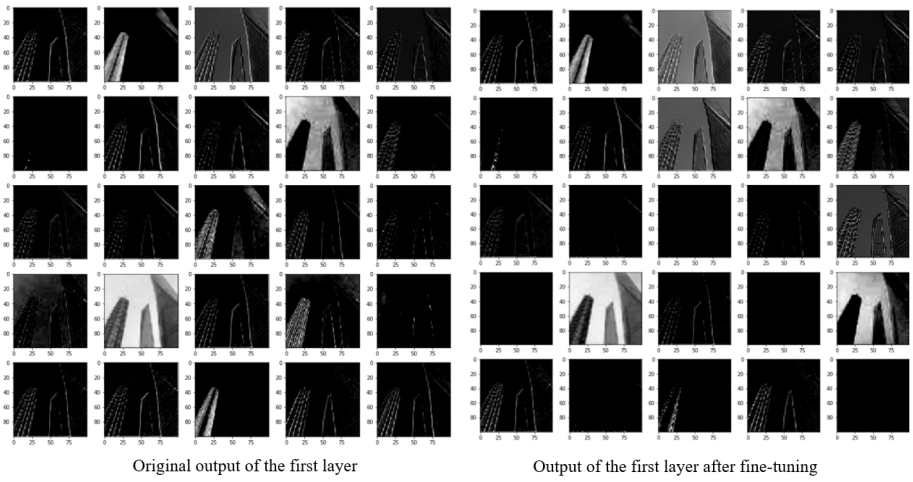


Fig. 6. Comparison of features from the first convolutional layer

6 Discussion

In this study, we demonstrated the substantial impact of the feature extractor in the transfer learning network performance and developed three modification methods to improve the learning ability of the feature extractor. We found that all three structural modifications can improve the feature extractor's performance. Given the different computational resources required, researchers can adjust the modification methods according to their conditions.

Cutting out the last layer or the last block can enhance the feature extractor performance. The results proved the feature evolution in the previous study [8]. The experiments indicate that simplifying the network properly (delete unappropriated features and save the appropriate ones) can improve the feature extractor's capability. Cropping more than three layers negatively affected the network significantly. It suggested the features from the shallower layer cannot represent the input properly. To our best knowledge, it is the first to use the simplifying strategy to enhance the feature extractor. The feature extractor simplifying strategy can reduce the required computational source for training. This optimization on the feature extractor is highly recommended when the computational resource is limited.

Fine-tuning the last convolutional layer can enhance the feature extractor performance maximally. This approach converted the extracted features to be more related to the target domain. Thus, feature processing could better exploit these extracted features. However, when an additional layer was fine-tuned, the test accuracy drops dramatically. When the last two layers were fine-tuned, more than 4 million parameters were made trainable. The limited training data cannot train such amount of parameters and overfitting occurred. The key is achieving a balanced trade-off between overfitting and fine-tuning. This method increases the computational cost significantly. This method is recommended when the feature extractor performance is the first priority.

Fine-tuning the first layers is a benefit to the feature extractor. By fine-tuning the first convolutional layers, we observed an improvement to the classification accuracy of 1.11%. The feature maps show the fine-tuned layers can extract more general features. In contrast to the mainstream perception, we proposed that even at the very first layer, the extracted features are already source-domain-oriented. Fine-tuning the first layers can center the extracted features to the target domain. Compared with another fine-tuning strategy, the computational cost is significantly reduced in this method. Therefore, we consider that low-level features extracted from the first part of the feature extractor can be fine-tuned more efficiently than high-level features. To all we know, this is the first study on fine-tuning the first layers of the feature extractor in a CNN-based transfer learning network. This provides researchers using similar networks an efficient method to enhance their networks.

The results indicate the extracted feature mismatching limits the feature extractor performance even then network based on the extractor is well-trained and well-performed. This study proposes three novel structural modifications to make the feature extractor extract features better represent the input. All three

modifications are direct modifications to the extractor structure. The modification implementations are straightforward. This study provides practical ways to enhance the feature extractor. Researchers applying transfer learning networks to practical applications can apply the findings to optimize their feature extractor.

7 Conclusion

In this work, the experimental results demonstrate that the structural modifications on the feature extractor can improve its performance. We found that fine-tuning the last convolutional layer of the feature extractor improved its performance more than the other two modifications. However, in this case, over two million parameters are fine-tuned, and this incurs high computational costs. Fine-tuning the first convolutional layer of the feature extractor improved classification accuracy by 1.11%. Here, only 1792 additional parameters are required for training. This optimization is very efficient. For the simplification strategy, the feature extractor could be enhanced without additional parameters. The experimental results confirm the importance of these feature extractor modifications. The results indicate the proposed structural modifications have the potential to enhance the feature extractors and transfer learning networks in practical applications. In the future, we will further investigate the relationship between transfer learning network performance and the degree of freedom on network parameters and training data.

References

1. Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
2. Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105, 2012.
3. Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.
4. Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.
5. Pei Cao, Shengli Zhang, and Jiong Tang. Preprocessing-free gear fault diagnosis using small datasets with deep convolutional neural network-based transfer learning. *Ieee Access*, 6:26241–26253, 2018.
6. Sijin Ren and Cheryl Q Li. Robustness of transfer learning to image degradation. *Expert Systems with Applications*, 187:115877, 2022.
7. Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2009.
8. Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? *arXiv preprint arXiv:1411.1792*, 2014.

9. Hui Liu, Jing Wu, Wenzhuo Lu, John A Onofrey, Yi-Hwa Liu, and Chi Liu. Noise reduction with cross-tracer and cross-protocol deep transfer learning for low-dose pet. *Physics in Medicine & Biology*, 65(18):185006, 2020.
10. Qingshan Liu, Renlong Hang, Huihui Song, and Zhi Li. Learning multiscale deep features for high-resolution satellite image scene classification. *IEEE Transactions on Geoscience and Remote Sensing*, 56(1):117–126, 2017.
11. Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning*, pages 6105–6114. PMLR, 2019.
12. Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.
13. John Bralich, Daniël Reichman, Leslie M Collins, and Jordan M Malof. Improving convolutional neural networks for buried target detection in ground penetrating radar using transfer learning via pretraining. In *Detection and Sensing of Mines, Explosive Objects, and Obscured Targets XXII*, volume 10182, page 101820X. International Society for Optics and Photonics, 2017.
14. Mohsen Ghafoorian, Alireza Mehrtash, Tina Kapur, Nico Karssemeijer, Elena Marchiori, Mehran Pesteie, Charles RG Guttmann, Frank-Erik de Leeuw, Clare M Tempany, Bram Van Ginneken, et al. Transfer learning for domain adaptation in mri: Application in brain lesion segmentation. In *International conference on medical image computing and computer-assisted intervention*, pages 516–524. Springer, 2017.
15. David Le, Minhaj Alam, Cham K Yao, Jennifer I Lim, Yi-Ting Hsieh, Robison VP Chan, Devrim Toslak, and Xincheng Yao. Transfer learning for automated octa detection of diabetic retinopathy. *Translational Vision Science & Technology*, 9(2):35–35, 2020.
16. Yuqing Gao and Khalid M Mosalam. Deep transfer learning for image-based structural damage recognition. *Computer-Aided Civil and Infrastructure Engineering*, 33(9):748–768, 2018.
17. Maxime Oquab, Leon Bottou, Ivan Laptev, and Josef Sivic. Learning and transferring mid-level image representations using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1717–1724, 2014.
18. Rahaf Aljundi and Tinne Tuytelaars. Lightweight unsupervised domain adaptation by convolutional filter reconstruction. In *European Conference on Computer Vision*, pages 508–515. Springer, 2016.
19. Boris Shirokikh, Ivan Zakazov, Alexey Chernyavskiy, Irina Fedulova, and Mikhail Belyaev. First u-net layers contain more domain specific information than the last ones. In *Domain Adaptation and Representation Transfer, and Distributed and Collaborative Learning*, pages 117–126. Springer, 2020.
20. Puneet Bansal. Intel image classification – image scene classification of multiclass.
21. Ahmed Hijab, Muhammad A Rushdi, Mohammed M Gomaa, and Ayman Eldeib. Breast cancer classification in ultrasound images using transfer learning. In *2019 Fifth International Conference on Advances in Biomedical Engineering (ICABME)*, pages 1–4. IEEE, 2019.
22. Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256. JMLR Workshop and Conference Proceedings, 2010.

23. Thomas M Cover. *Elements of information theory*. John Wiley & Sons, 1999.
24. Tijmen Tieleman, Geoffrey Hinton, et al. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 4(2):26–31, 2012.

Creating Customers That Never Existed
Synthesis of E-commerce Data Using CTGAN

Melle Mendikowski[1] and Mattis Hartwig[1,2]

[1] German Research Center for Artificial Intelligence, Lübeck
[2] singularIT GmbH, Leipzig

Abstract. Various e-commerce use cases that companies implement in applications rely on personal data of customers. Privacy and data protection play an important role when discussing the usage of personal customer data resulting in a conflicting demand between data collection and data protection. Researchers have found a promising solution to this problem: the generation of synthetic data which is not connected to real people. In this paper, we use the deep learning architecture Conditional Tabular Generative Adversial Network (CTGAN) to synthesize e-commerce data. Especially the categorical relationships between columns of e-commerce data include fixed dependencies, where e.g. an entry in the sub-category column is defining the entry in the category column as well. These specific characteristics result in the necessity to evaluate the suitability of the CTGAN architecture for synthesizing e-commerce data which is the focus of this paper. We present a new similarity measure for synthetic and original datasets that focuses on categorical correlations: the Cramer's V deviation (*CV-deviation*). In our experiments, we create synthetic e-commerce data from a publicly available dataset using CTGAN. We use an existing and our newly developed *CV-deviation* measure in hyperparameter selection and compare the outcomes. By incorporating *CV-deviation* into the performance metric, we manage to increase the ability of CTGAN to preserve correct categorical relations by 63%. Despite the enhancements the evaluation of the synthetic datasets shows that there is still room for improvement of the overall architecture because it seems difficult for the CTGAN model to efficiently learn all categorical constraints automatically.

Keywords: CTGAN · Categorical Relations · E-Commerce · *CV-deviation*.

1 Introduction

One of Amazon's biggest success factors has been its personalized recommendation system, which is based on massive amounts of data involving transactions from millions of customers [27]. Recommendation systems in general need a lot of data to learn the relationships between products, preferences and customer segments. Not only in the recommendation systems sector, but also in many other e-commerce areas, the analysis of customer data is a key element, e.g. in

detection of fraud, in forecasting and inventory management. To solve the increasing demand new ways and models to create e-commerce data are urgently needed [17]. Due to its personal and sensitive nature, handling customer data brings its own challenges: How can people's personal data be protected, when it is used for analysis [7]? How can sensitive data be shared and multiplied? A promising solution to this problems is synthetic data: The generation of new data containing as many properties and information of the original data as possible while not being linked to the same individuals present in the original dataset [2]. Synthetic data research is an important area that is becoming increasingly popular throughout the machine learning field [18].

Especially in Europe, with the new data protection law GDPR, research achievements in synthetic e-commerce data are of central importance: The GDPR has led to a decline in usable data and seems to affect the revenues of the European e-commerce platforms [8]. The ability to use synthetic data to preserve information content while effectively protecting customer privacy could mean a breakthrough in European e-commerce market. Especially smaller players who are uncertain about the risks of using customer data, could benefit from the usage and sharing of synthetic data.

One architecture that is widely known due to numerous successes in generating so-called "deep fakes", i.e. deceptively real synthetic images, videos or audio content, is the Generative Adverisal Network (GAN) [24]. The Conditional Tabular Generative Network (CTGAN), a specialization of the GAN architecture for synthesizing tabular data was presented in 2019 by Lei Xu et al. in 'Modeling Tabular Data Using Conditional GAN' [26]. A first evaluation of the CTGAN in generating synthetic insurance data focusing on datasets with scalar values achieved promising results [14].

In this paper, we address the question of whether the CTGAN architecture can provide promising results in the area of synthetic e-commerce data. E-commerce data contains many columns with categorical data such as product categories or postal codes. The correlations between products and other columns with categorical data is central to recommender systems [20].

We evaluate synthetic e-commerce datasets along several dimensions and pay close attention to the maintenance of important relationships between the columns with categorical data. The similarity of categorical correlations between synthetic and original datasets is measured by the newly proposed measure Cramer's V Deviation (*CV-deviation*).

In our experiments, we generate synthetic data from a publicly available e-commerce dataset using CTGAN and implement a grid search that optimizes a subset of CTGAN's hyperparameters in respect to the e-commerce data. Additionally to evaluating a base approach, we include the *CV-deviation* measure in the hyperparameter training of the CTGAN model to investigate if this change of focus in the hyperparameter training has an effect on the evaluation parameters. By incorporating *CV-deviation* into the performance metric of the hyperparamter training, we can increase the the average categorical integrity of the synthetic e-commerce dataset by 17 percentage points.

The following Section 2 contains preliminaries on the CTGAN architecture. Subsequently, we present related work to this paper in Section 3. Section 4 displays our method including the formula for *CV-deviation* and introduces our implementation process. Section 5 discusses the evaluation of the synthetic e-commerce datasets. Lastly, Section 6 presents the central findings of this paper and points out further research directions.

2 Preliminaries

The GAN architecture was published in 2014 by Ian Goodfellow and his team, it consists of two artificial neural networks, the generator G and the discriminator D, which resemble two players playing a minmax game against each other [9]. The generator G produces synthetic data samples of a desired instance from a random noise source. Alternating with original samples from the real data distribution, these synthetic samples are fed into the discriminator network D, which determines whether the input belongs to the real dataset. During training the generator learns to create more realistic instances, while the discriminator tries to identify those generated instances with greater accuracy [9].

The CTGAN, whose architecture is illustrated in Figure 1, consists of two neural networks a generator G and a critic C that corresponds to the discriminator in the classic GAN architecture. The CTGAN's critic scores either 10 real or generated data series according to the network's estimated authenticity of the data. There are two main innovations adapted to the generation of synthetic tabular data: mode specific normalization and a conditional training by sampling [26].

Mode specific normalization is used to transfer the values of the continuous columns into a combination of a scalar value and a one-hot vector that increases the ability of CTGAN to create continuous columns with multiple modes during generation process. Figure 1 shows two datasets as input to the critic, one consisting of 10 rows of real data samples and the other consisting of 10 synthetic data rows. The continuous values of each row in those sets are represented using a mode-specific normalization and their categorical values are represented as one-hot vectors.

Another problem with GANs when creating tabular data is the highly imbalanced categorical columns, i.e., a column that consists of 90 % of one major category. CTGAN's conditional training approach applies a specific column with categorical data and a specific category from that column as a constraint to the data generation and the sampling process. Through an integration in the loss function, CTGAN learns to implement this condition during training. Figure 1 illustrates the influence of this condition as a conditional vector for the generator G and as a sampling filter for the input of the critic C. The column that is affected by this condition is chosen at random. To choose the category of the selected column as a condition, a probability mass function is calculated in which the calculated probability mass of each category is the logarithm of the frequency of that category in the selected column. To ensure that the infrequent

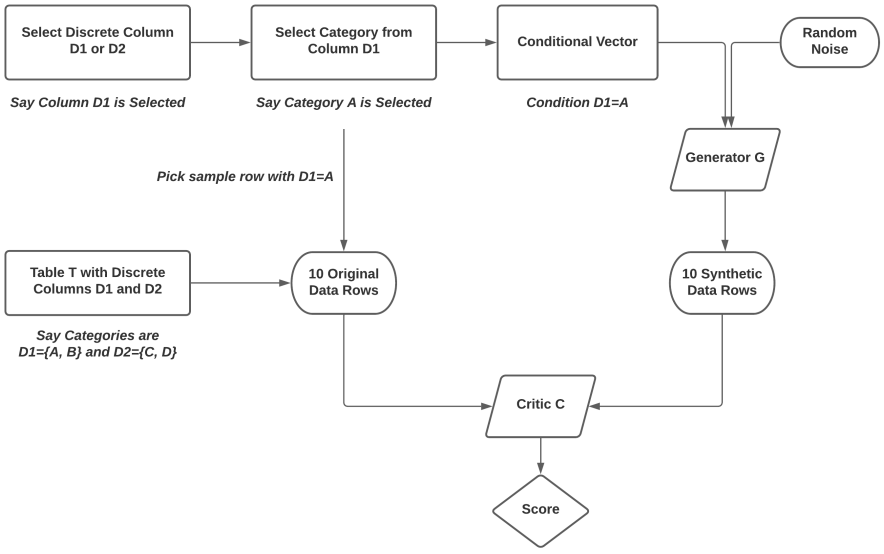


Fig. 1: An illustration of CTGAN architecture.

categories of this selected column are considered to a larger extent, the category for the condition is randomly determined from this calculated probability mass function.

In addition to these enhancements, CTGAN also utilizes recent advances in GAN training such as WGAN-GP, an improved Wasserstein GAN with gradient penalty [10].

3 Related Work

Since its publication in autumn 2019, CTGAN has been combined with other architectures and evaluated with different datasets. Rosenblatt et.al. (2020) made the architecture differentially private, a formalization of privacy [6], by combining CTGAN with DP-SGD and PATE technique [19]. Similar to the DPGAN approach [25], applying DP-SGD to CTGAN adds random noise to the discriminator and prunes the norm to achieve differential privacy [19]. The PATE-CTGAN approach is also inspired by a similar approach on the GAN architecture [12], the original dataset is divided into subsets and each of the subsets has its own generator and discriminator network [19].

CTGAN has been evaluated on other types of table data and achieved promising results. Similar to our approach, Kuo et al. made small changes to the original CTGAN architecture that promise advantages for generating insurance datasets [14]. Their workflow involves using true data frequency instead of log-frequency when choosing a value for the condition that influences the generator and the

sampling process. In 2021, Min Jong et.al. evaluated the CTGAN and the TGAN in their ability to generate synthetic EEG data. In their evaluations, the CTGAN achieved higher similarity scores than the TGAN, while the machine learning performance of the two generative architectures remained similar [15]. In order to obtain better training data for the evaluation of stability of power systems, Han et al. use an approach that first creates tabular data with CTGAN, which is then further processed. The data created with this framework is analyzed with different methods and achieves good values in several metrics [11]. None of the above CTGAN evaluations focus especially on relationships between columns with categorical data, which are in particular important in e-commerce data where, for example, products are divided into different categories that must be preserved in the synthetic data.

4 Dataset

For our synthesis of e-commerce data, we use the "Superstore" dataset that contains data on purchases from 2014 to 2017 from an U.S. online store with various offerings ranging from books to furniture or other household items. To increase transparency, we choose a dataset which is publicly available on the Kaggle platform. The dataset does not appear to be anonymized [23].

The unprocessed "Superstore" dataset consists of 9,994 rows and 20 columns with different information. In order to use the "Superstore" dataset for synthesis, some information duplicated with the respective ID such as *Product Name* and *Customer Name* are deleted. The "Superstore" dataset has 13 columns with categorical data, the remaining 4 columns are continuous. The dataset includes 793 individual customers who ordered 1862 different products in 5,009 individual orders. All products can be divided into 17 subcategories, which in turn are divided into the three categories *Furniture*, *Office Supplies* and *Technology* [23].

5 Method and Implementation

In this section, we describe our method and the steps in our implementation. We start by introducing *CV-deviation*, our new evaluation metric for synthetic datasets to better incorporate columns with categorical data. We then display the evaluation methods we use to analyze the synthetic e-commerce datasets. Furthermore, we describe the general implementation of training the CTGAN models. Lastly, we present our technical setup and performed grid search.

5.1 Cramer's V Deviation

In order to achieve higher similarity between synthetic and original e-commerce data, we intend to use a performance metric that especially supports categorical integrity of synthetic data. The Cramer's V is a measure of statistical association between two categorical variables, it returns a value between 0 and 1,

with a higher value representing a greater correlation of the two variables. The Cramer's V with Wicher Bergsma correction (CV) for column pair (D_i, D_j) with categorical data and number of categories $|D_i|$ and $|D_j|$ in table T with Number of rows N_r is calculated as follows [4]:

$$\begin{gathered} CV = \sqrt{\frac{\tilde{\Phi}^2}{min(\tilde{k}-1, \tilde{r}-1)}}, \\ \tilde{\Phi}^2 = max(0, \Phi^2 - \frac{(k-1)(r-1)}{(n-1)}), \;\; \Phi = \frac{\chi^2}{n}, \\ \tilde{k} = k - \frac{(k-1)^2}{n-1}, \;\; \tilde{r} = r - \frac{(r-1)^2}{n-1}, \\ \chi^2 = \text{chi-square test of independence [13] of } (D_i, D_j), \\ k = |D_i|, \;\; r = |D_j|, \;\; n = N_r \end{gathered} \tag{1}$$

We define the corrected Wicher Bergsma Cramer's V of a column pair with categorical data $j \in \mathcal{P}_2(D_1, ..., D_{N_d})$ of tabular dataset T with columns with categorical data $D = \{D_1, ..., D_{N_d}\}$ as: $CV_T(j)$.

To create a performance metric using CV as a base, we combine the Cramer's V with Wicher Bergsma correction with the Root Mean Squared Error [5] and obtain the Cramer's V Deviation (*CV-deviation*). The *CV-deviation* of real table T and synthetic table T_{syn} with columns with categorical data $D = \{D_1, ..., D_{N_d}\}$ and $(|D| > 1)$ is calculated as follows:

$$\textit{CV-deviation}(T, T_{syn}) = \sqrt{\frac{1}{|\mathcal{P}_2(D)|} \sum_{j \in \mathcal{P}_2(D)} (CV_T(j) - CV_{T_{syn}}(j))^2} \tag{2}$$

The *CV-deviation* measures the difference of the statistical correlations between all pairs from the columns with categorical data in the synthetic table T_{syn} to the corresponding correlations in the real table T. The *CV-deviation* is a similarity measure for a pair of real and synthetic tabular data and therefore can only be applied to datasets that have the same columns with categorical data. The *CV-deviation* can take 0 as the lowest result and 1 as the highest value, the closer the result is to 0, the more similar are the statistical relationships among the columns with categorical data with respect to the CV value. If we were to calculate the *CV-deviation* from a dataset to itself, the result would be 0.

5.2 Evaluation Method

We evaluate the synthesized e-commerce datasets in detail and compare them with the original dataset. Therefore, we examine the similarity of column distributions of synthetic datasets to the column distributions of the original "Superstore" dataset looking at overall distribution measures, upper and lower bounds for continuous columns and number of categories for columns with categorical data. We also inspect the integrity of relationships between column pairs with

categorical data in our synthetic datasets. Some columns with categorical data in e-commerce data have a special relationship to each other that does not allow new combinations in the synthetic data, i.e., a cellphone always belongs to the sub-category technology and not furniture. Detection of column pairs with this categorical integrity requires expert knowledge about the relationships between columns, which is not always available. We furthermore compare the Cramer's V values of the synthetic datasets with the results of the original dataset to display an overview of the similarity of categorical statistical correlations of the synthetic datasets to the original dataset.

5.3 General Implementation

The CTGAN training process and all evaluation of the synthetic datasets is written in Python 3.7. We create CTGAN models using version 0.12 of the Synthetic Data Vault (SDV) library. The SDV library is an set of open source software systems concerning synthetic data, this project was launched by the Massachusetts Institute of Technology in 2018 [21]. The implementation of the SDV CTGAN is the realization of the original paper [26]. For each combination of the selected hyperparameters that we optimize in this paper, we create a CTGAN model that is trained with the appropriate parameters on the "Superstore" dataset. After training, we save every CTGAN model and create 10,000 rows of synthetic data with the saved model.

5.4 Gridsearch and Technical Setup

For optimizing CTGAN in respect to e-commerce data we implement a grid search over the following hyperparameters: epochs {100, 300, 500, 700, 900}, batch size {100, 300, 500, 700, 900, 1000}, log frequency {True, False} (whether to use the logarithm of the frequency of a value in a column with categorical data to determine the conditional input), learning rate (LR) for the critic {2e-4, 2e-5} and critic steps {1,5} (number of critic updates to do for each generator update). The original CTGAN uses 1 critic update and the default from the WGAN-GP paper is 5 [10]. The grid search results in 240 different CTGAN models. We compute each model on a Quadro RTX 6000 and parallelize this process multiple times on a cluster server.

Each of the 240 created synthetic datasets is evaluated with two performance metrics. The first performance metric is the SDV Single Table Metric, which is included in the SDV library. The SDV Single Table Metric itself is a collection of other lower level metrics that can be divided into multiple groups. We use the following three groups of metrics (based on the SDV framework) to measure the quality of our synthetic data:

statistical metrics: KSTest, CSTest
likelihood metrics: GMlikelihood
detection metrics: LogisticDetection

The SDV Metric returns a score between 0 and 1, being 0 the worst and 1 the best possible score [21].

As a second performance metric, we supplement the SDV metric with an additional component for categorical relationships: we use a combined and equally weighted score from the normalized SDV metric value and the normalized 1 - CV deviation value. This combined metric, CVSDV, also scores the synthesized datasets on a scale of 0 to 1, with a score closer to 1 indicating higher similarity to the original dataset.

6 Results and Discussion

In this chapter, we evaluate two synthetic datasets, the dataset that scores highest in the SDV metric and the dataset that has the highest score in our new CVSDV metric. We start by evaluating the similarity of the synthetic column distributions to the original distributions. Afterwards, we inspect how closely the original relationships of the columns with categorical data are transferred to the synthetic data. The two best combination of hyperparameters in terms of SDV metric and CVSDV metric are shown in Table 1.

Table 1: Hyperparameters synthetic datasets with highest SDV or CVSDV

Dataset	Epochs	Batch Size	Log Freq.	Cr. Steps	Cr. LR	SDV	CVSDV
highest SDV	100	1000	False	5	2e-4	0.6323	0.6477
highest CVSDV	500	900	False	5	2e-4	0.5835	0.7945

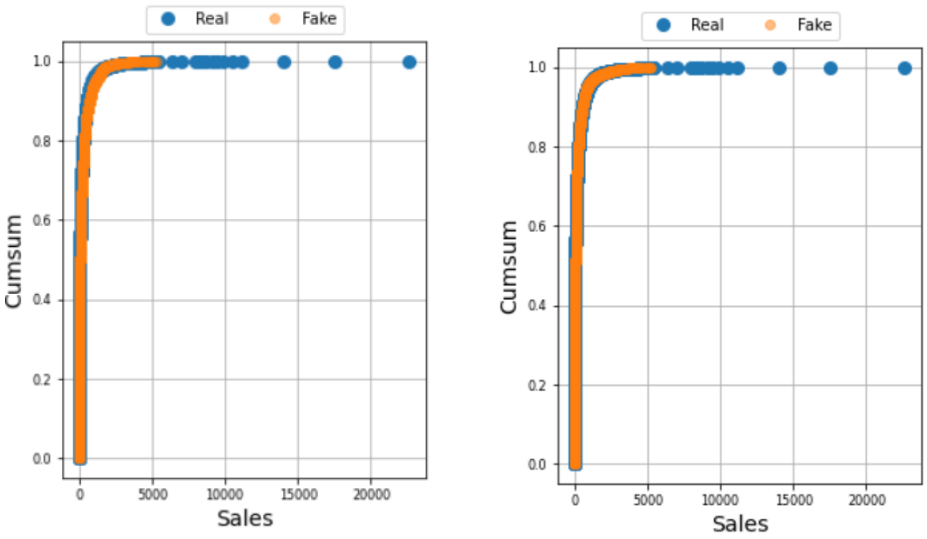
6.1 Column Distribution

Both the Kolmogorov–Smirnov test [3] for continuous columns and the Chi-Squared test [13] for columns with categorical data show high similarity between synthetic and original dataset (see Table 2).

Table 2: Chi-Squared test and Kolmogorov–Smirnov test results.

Synthetic Dataset	Chi-Squared Test	Kolmogorov–Smirnov test
highest SDV score	0.998794	0.880171
highest CVSDV score	0.995688	0.908547

All continuous columns of the two synthetic datasets do not exceed the value range of the original columns. However, it is noticeable that both synthetic datasets strongly decimate the upper and lower limit of the value range in some continuous columns like *Sales*. The table-evaluator [22] Figure 2 shows the synthetic datasets cumsum of the *Sales* column, a statistical quality control that



(a) Dataset highest SDV score.

(b) Dataset highest CVSDV score.

Fig. 2: Cummulative sum *Sales* column in synthetic datasets.

measures change [1]. We recognize a few purchases of more than 5,000$ in the *Sales* distribution of the original dataset, these edge cases are missing in both synthetic datasets resulting in a about 76% smaller value range. Overall both synthetic datasets lack about 40% value range in continuous columns compared to the original dataset.

Some synthetic columns have fewer categories than the columns in the original dataset. For example, this is the case for columns like *Product ID* that have a large amount of possible categories (1862). Columns with fewer categories like *Region* or *Sub-Category* contain the same categories as the original dataset. Figures 3, 4a and 4b display the distribution of the column with categorical data *Sub-Category*: a strong increase of purchases of "Bookcases" can be seen in both synthetic datasets. The cause for these unusually high "Bookcases" values could be mode collapse, a failure typical for GAN architectures [26].

6.2 Categorical Integrity

Figures 5, 6a and 6b illustrate the Wicher Bergsma Cramer's V (CV) [4] values of all pairs of columns with categorical and temporal data in the original dataset and in the two synthetic datasets. In the heatmaps, a high CV score indicating high statistical association is connected with a lighter color. The original dataset has high CV values between columns in the lower left quarter of the heatmap, this pattern is more evident in the heatmap of the dataset whose hyperparameters were optimized with the CVSDV metric. The best SDV score dataset has overall

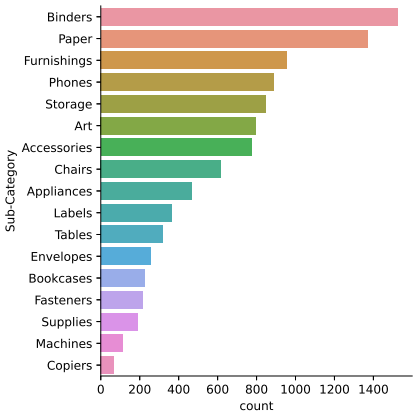


Fig. 3: Distribution *Sub-Category* column original dataset.

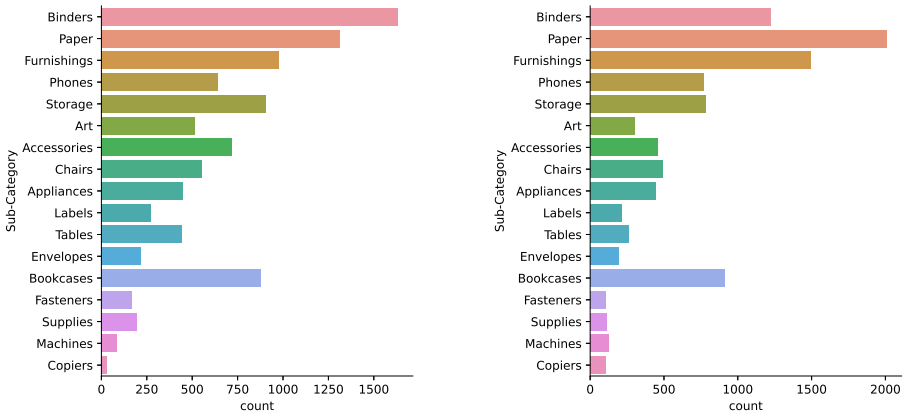
much lower CV values, visualized by the lower maximum value of the scale of the heatmap at 0.25, compared to the maximum value of the scale of the CVSDV dataset heatmap, which reaches a value of 0.6. The *CV-deviation* value confirms a closer proximity of the best CVSDV dataset to the original dataset, the *CV-deviation* of the best CVSDV dataset is 0.32 and that of the best SDV dataset reaches a higher value of 0.37.

The original CV heatmap (Figure 5) visualizes some special categorical relationships that do not allow new combinations in the synthetic dataset with very high values, such as *City* and *Postal Code* with a CV value of 0.99. In Table 3 we can see the absolute numbers and percentage by which both synthetic datasets correctly reflect these type of relationships. The synthetic dataset, whose model

Table 3: Categorial integrity in synthetic datasets.

Column Pair	Number Combinations	Correct SDV	Correct CVSDV
(*Category*/*Sub-Category*)	17	5,315(53%)	7,865(79%)
(*Category*/*Product ID*)	1,862	4,385(44%)	4,442(44%)
(*Product ID*/*Sub-Category*)	1,862	864(9%)	1,007(10%)
(*City*/*State*)	604	1,268(13%)	3,099(31%)
(*City*/*Postal Code*)	632	489(5%)	1,643(16%)
(*City*/*Region*)	583	4,124(41%)	6,506(65%)
(*State*/*Postal Code*)	631	1,015(10%)	2,546(25%)
(*State*/*Region*)	49	3,589(36%)	6,968(70%)
(*Region*/*Postal Code*)	631	2,982(30%)	5,226(52%)

was optimized with the CVSDV metric, achieves a higher number of correct matches for each individual column pair. The largest absolute difference occurs at (*State*/*Region*), here the CVSDV dataset achieves a better result by



(a) Dataset highest SDV score. (b) Dataset highest CVSDV score.

Fig. 4: Distribution *Sub-Category* column in synthetic datasets.

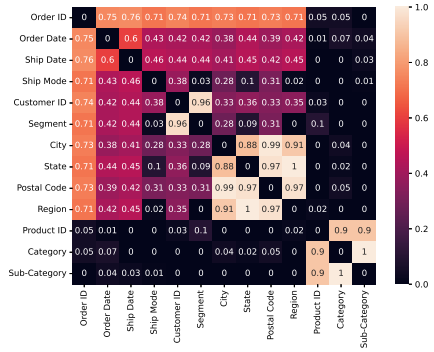
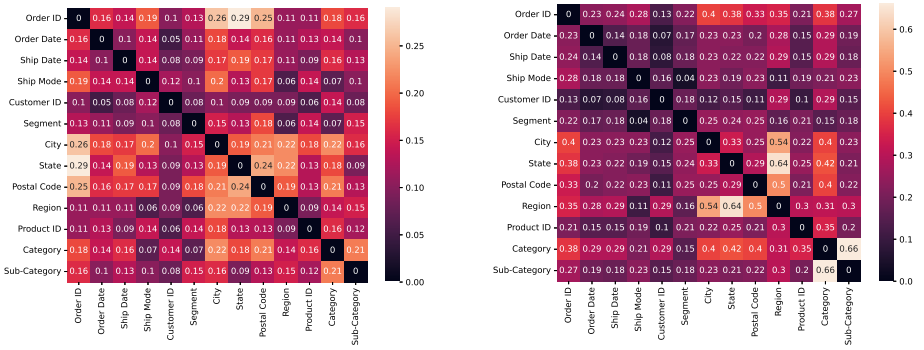


Fig. 5: Cramer's V of column pairs in original dataset.

3379 correct data rows, which means an increase of 94 % compared to the SDV dataset. Applying the CVSDV performance metric increases the number of correctly assigned rows for the column pair (*City*/*Postal Code*) by as much as 235%, which is the highest percentage improvement. It is noticeable that for categories with many possible combinations like (*Product ID*/*Sub-Category*) both synthetic datasets achieve very low matches. Overall, the SDV dataset achieves an average of 27 % correct matches and the CVSDV dataset average is 17 percentage points (or 63%) higher at 44 % correct categorical assignments.

For completeness, we briefly consider the temporal integrity of the synthetic datasets. There are also temporal requirements that must be met in the synthetic data in order to reflect a real purchase process, e.g. the *Ship Date* must be temporally after the *Order Date*. The correct chronological order of the date columns



(a) Dataset highest SDV score. (b) Dataset highest CVSDV score.

Fig. 6: Cramer's V of column pairs in synthetic datasets.

is maintained in both datasets to 50 percent, the CVSDV dataset reaches only minimally better values than the SDV dataset.

7 Conclusion

Overall, the CTGAN architecture seems to be a promising architecture to generate e-commerce data. Both synthetic datasets have similar column distributions to the original dataset and the reduction of the definition range in continuous columns only plays a minor role, since this only affects a small subset of the data.

For a real world application of synthetic e-commerce data, it is important that each data row reflects a correct buying processes, and therefore keeping correct categorical relationships is a key point. For both evaluated synthetic datasets, this categorical integrity is only maintained at an average percentage of less than 50 percent: SDV metric dataset (27%) and CVSDV metric dataset (44%), which is not satisfactory. Especially for column pairs with a large number of categories, CTGAN has problems to reflect their relationships correctly in the synthetic data. However, there is a significant overall increase in the dataset whose CTGAN hyperparameters are optimized with the CVSDV metric. Applying the CVSDV performance metric more than doubles the number of correct assignments for some column pairs and improves the average categorical integrity by 17 percentage points.

In order to use CTGAN for the production of synthetic e-commerce data, other approaches are still needed that will lead to better categorical integrity. One approach, could be to integrate statistical evaluation metrics, such as the presented *CV-deviation*, into the direct training process of CTGAN and thus enforce greater adherence to categorical relations at an earlier stage. Another interesting approach could be increasing the pac size, i.e., the number of data rows that the critic receives as samples, to more than 10. Viewing multiple rows

of data simultaneously could make the correlations between columns more visible to the network and improve the ability of the CTGAN architecture to translate such relationships into synthetic data. To improve the overall performance of the CTGAN architecture other loss function could be tested which lead to a good performance in current GAN models like the adversial loss used in NSGAN with R1 regularization [16].

Another research direction would be to add the training of a real-life recommender system as a subsequent evaluation step. The recommender performance achieved with the synthetic dataset could be then compared with the recommender perforamance of the original dataset.

References

1. Barnard, G.A.: Control charts and stochastic processes. Journal of the Royal Statistical Society: Series B (Methodological) **21**(2), 239–257 (1959)
2. Bellovin, S.M., Dutta, P.K., Reitinger, N.: Privacy and synthetic datasets. Stan. Tech. L. Rev. **22**, 1 (2019)
3. Berger, V.W., Zhou, Y.: Kolmogorov–smirnov test: Overview. Wiley statsref: Statistics reference online (2014)
4. Bergsma, W.: A bias-correction for cramér's v and tschuprow's t. Journal of the Korean Statistical Society **42**(3), 323–328 (2013)
5. Chai, T., Draxler, R.R.: Root mean square error (rmse) or mean absolute error (mae)?–arguments against avoiding rmse in the literature. Geoscientific model development **7**(3), 1247–1250 (2014)
6. Dwork, C.: Differential privacy: A survey of results. In: International conference on theory and applications of models of computation. pp. 1–19. Springer (2008)
7. Gahi, Y., Guennoun, M., Mouftah, H.T.: Big data analytics: Security and privacy challenges. In: 2016 IEEE Symposium on Computers and Communication (ISCC). pp. 952–957 (2016). https://doi.org/10.1109/ISCC.2016.7543859
8. Goldberg, S., Johnson, G., Shriver, S.: Regulating privacy online: The early impact of the gdpr on european web traffic & e-commerce outcomes. Available at SSRN 3421731 (2019)
9. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. Advances in neural information processing systems **27** (2014)
10. Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., Courville, A.C.: Improved training of wasserstein gans. In: NIPS (2017)
11. Han, G., Liu, S., Chen, K., Yu, N., Feng, Z., Song, M.: Imbalanced sample generation and evaluation for power system transient stability using ctgan. In: International Conference on Intelligent Computing & Optimization. pp. 555–565. Springer (2021)
12. Jordon, J., Yoon, J., Van Der Schaar, M.: Pate-gan: Generating synthetic data with differential privacy guarantees. In: International conference on learning representations (2018)
13. Koch, G.G., Wiener, L.E.: Chi-squared tests: Basics. Wiley StatsRef: Statistics Reference Online pp. 1–20 (2014)
14. Kuo, K., et al.: Generative synthesis of insurance datasets. Tech. rep. (2020)

15. Lee, J.S., Lee, O.: Ctgan vs tgan? which one is more suitable for generating synthetic eeg data. Journal of Theoretical and Applied Information Technology **99**(10) (2021)
16. Mescheder, L., Geiger, A., Nowozin, S.: Which training methods for gans do actually converge? In: International conference on machine learning. pp. 3481–3490. PMLR (2018)
17. Moorthi, K., Dhiman, G., Arulprakash, P., Suresh, C., Srihari, K.: A survey on impact of data analytics techniques in e-commerce. Materials Today: Proceedings (2021)
18. Nikolenko, S.: Synthetic data in deep learning. In: School-conference "Approximation and Data Analysis 2019". p. 21 (2019)
19. Rosenblatt, L., Liu, X., Pouyanfar, S., de Leon, E., Desai, A., Allen, J.: Differentially private synthetic data: Applied evaluations and enhancements (2020)
20. Sarker, I.H.: Machine learning: Algorithms, real-world applications and research directions. SN Computer Science **2**(3), 1–21 (2021)
21. Sdv - the synthetic data vault — sdv 0.12.1 documentation. https://sdv.dev/SDV/, accessed: 2021-8-13
22. table-evaluator: A package to evaluate how close a synthetic data set is to real data. https://pypi.org/project/table-evaluator/, accessed: 2021-12-13
23. Us superstore data. https://www.kaggle.com/juhi1994/superstore, accessed: 2021-09-30
24. Whittaker, L., Kietzmann, T.C., Kietzmann, J., Dabirian, A.: "all around me are synthetic faces": The mad world of ai-generated media. IT Professional **22**(5), 90–99 (2020)
25. Xie, L., Lin, K., Wang, S., Wang, F., Zhou, J.: Differentially private generative adversarial network. arXiv e-prints pp. arXiv–1802 (2018)
26. Xu, L., Skoularidou, M., Cuesta-Infante, A., Veeramachaneni, K.: Modeling tabular data using conditional gan. Advances in Neural Information Processing Systems **32**, 7335–7345 (2019)
27. Zakir, J., Seymour, T., Berg, K.: Big data analytics. Issues in Information Systems **16**(2) (2015)

Risky Tackle Detection from American Football Practice Videos using 3D Convolutional Networks

Nasik Muhammad Nafi[1], Scott Dietrich[2], and William Hsu[1]

[1] Kansas State University, Manhattan, KS 66502, USA
{nnafi,bhsu}@ksu.edu
[2] Barry University, Miami Shores, FL 33161, USA
sdietrich@barry.edu

Abstract. In this paper, we introduce the problem of risky tackle detection from American football practice videos and propose a 3-stage Convolutional Neural Network (CNN)-based pipeline to improve detection accuracy. At first, we propose an anomaly detection-based approach to temporally localize the tackle action. Spatial regions of interest are then identified using an object recognition model. Finally, 3D convolution is applied to classify risky and safe tackles based on spatiotemporal features. Our approach trades off between end-to-end action classification from untrimmed videos and precise localization of temporal anchors of an action. We conduct our experiment on a newly created data set that contains 178 annotated videos collected from seven different practice fields. We empirically demonstrate that our proposed method outperforms state-of-the-art video classification and anomaly detection approaches applied directly to untrimmed tackle videos.

Keywords: American Football, Head Injury, Risky Tackle Identification, Deep Learning, Sports Video Classification.

1 Introduction

In this work, we address the problem of simultaneous action detection and risk estimation as a classification task for videos. Such computer vision applications in sports span a gamut of static scene analysis to high frame rate videos covering entire practice sessions, and from brief training exercises to plays. The key rationale for visual analysis of sports videos is monitoring and then providing early warnings of potentially injurious practices. This would allow coaches to intervene to prevent injury and mitigate resulting risk and harm, whether physical, psychological, or financial, from improper tackling. Specifically, the Centers for Disease Control (CDC) estimates that between 1.6 and 3.8 million sports-related concussions (SRC) are reported annually with American football showing the highest proportion of head injuries or concussions among all sports [21] [5]. Research shows that in youth football, on an average, one player out of every 33 players may suffer a concussion during the season. Concussions occur at a rate

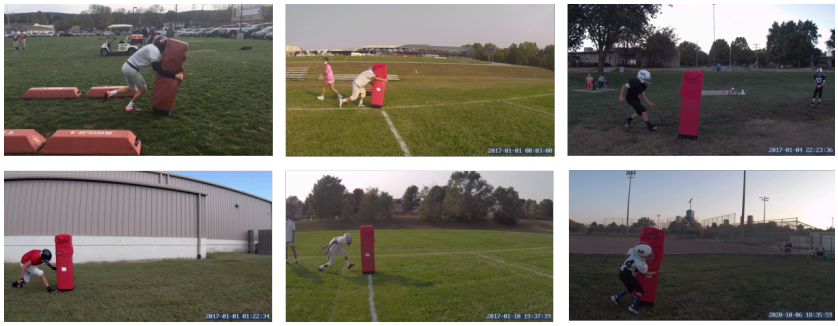


Fig. 1: Representative frames from videos collected at different practice fields.

of 9.9 per 10,000 athlete exposures; where each athlete exposure is considered one play either in practice or in a game [22]. In addition, head impact may cause brain injuries such as hemorrhage, hematoma, and edema. Annually, millions of dollars are spent to treat injured players and maintain reserved players [36]. Furthermore, this adversely affects the teams, both in competitive performance and reputation.

Two-thirds of all football-related head injuries occur during practice and one-third during games, 47% of all SRC occur as a result of head-to-head collisions [4]. Researchers have found that early exposure to American football may have a long-term neuropsychiatric and cognitive effects such as Chronic Traumatic Encephalopathy (CTE) due to repeated head impacts [29] [1]. Learning proper tackle form at an early age is an important developmental milestone for reducing unnecessary head impacts among youth football players [21] [23].

Identification and correction of improper-tackle techniques is a key step for establishing a safe playing environment. Coaches wanting to reduce the potential for player to player head impacts may choose to use blocking dummies when teaching the skill to young players. Practice tackles are filmed so that athletic trainers and coaches can identify dangerous postures and provide corrective feedback on player performance. However, these video assessments are carried out manually by human judges [28] [39] [19]. Manual processing of the videos to classify risky or safe tackle requires a substantial amount of effort and time from human assessors.

CNN-based architectures have been shown to be successful at extracting novel visual features directly from RGB images [13] [38]. Use of CNN in tandem with Long Short-Term Memory (LSTM) network and the introduction of 3D convolution have made a breakthrough in many video processing tasks such as activity recognition, event or action localization, anomaly detection [17] [8] [40] [16] [3] [37]. This influenced researchers to adopt deep learning-based computer vision approaches in sports analytics [10] [31]. However, the inherent differences in actions performed in different sports pose a different set of challenges.

Automatic detection of the risky forms of tackle solely based on videos can greatly improve a coach's ability to correct player behavior and reduce the likelihood that the players sustain head impacts. More importantly, this will help the players to find out the overall safety ratings of their tackles just after performing them rather than waiting for more than a week while the coaches analyze the videos. To the best of our knowledge, no prior research has attempted to classify tackles from videos of American football practice. In our work, we first exploit an anomaly detection mechanism to temporally segment the informative frames containing the tackle and then leverage an existing state-of-the-art object detection model to extract regions of interest from those frames. In last stage, a customized 3D ConvNet is used to classify risky and safe tackles from the spatiotemporally segmented frame sequence.

To summarize, our key contributions are as follows:

- We introduce the task of risky tackle detection directly from videos of American football practice with a tackle dummy.
- We present a set of 178 labeled American football tackle practice videos collected in the United States.
- We propose a framework for detecting risky tackles from practice videos using only video-level annotation.
- We conduct a comparative analysis of our proposed pipeline with state-of-the-art video classification and anomaly detection approaches for untrimmed video and evaluate the results in terms of precision, recall, and F1-score.

2 Related Work

Video Classification: One of the core tasks of video processing is video classification, commonly referred as activity recognition. In the last few decades, it was very common to use hand-crafted features for video representation. Spatiotemporal interest points (STIPs) [27], 3D variants of scale-invariant feature transform (SIFT-3D) [33], and histogram of oriented gradients (HOG-3D) [24], improved Dense Trajectories (iDT) [41] demonstrated promising results. Recent CNN-based approaches have already gained success over those hand-crafted features [17] [8] [40]. One common approach is to extract frame-level features using 2D convolution followed by Long Short-Term Memory (LSTM) cells to capture temporal dynamics from those frame-level features [8]. 3D ConvNet eliminates the need for LSTM blocks by extending 2D convolution into the temporal dimension, making it well-suited for spatiotemporal feature learning directly from video [40] [16].

Two-stream networks [35] [3] utilize both RGB frames and optical flow frames. Optical flow can capture apparent motion information invariant to appearance. The RGB and flow frames are fed into identical ConvNet to extract features and are fused at some particular stage. C3D [40], I3D [3], R(2+1)D have proved that a video classification network trained on a sufficiently large data set such as

Kinetics[18], Sports-1M [17] can be used to extract video features for completely different tasks from other domains.

A different approach, however, is to consider binary video classification as an anomaly detection problem. Most anomaly detection approaches use unsupervised or semi-supervised methods such as dictionary learning [45], topic modeling [15], histograms [6], or autoencoders [44] to learn the distribution of normal video, so it can distinguish the anomalies. Some recent approaches attempt to solve the problem with supervised learning using both normal and anomalous videos with video-level annotation [37].

Action Localization: Most techniques for action localization assume that untrimmed input videos are annotated with the temporal anchor of the action. They then treat the task as an iterated image classification task, where the system needs to classify each candidate window derived from running a temporal sliding window over the whole video [30] [10]. More recently, to reduce the number of candidate windows, temporal action proposals [9] [14] have been introduced. Buch et al. [2] presented a single-stream temporal action proposal (SST) to mitigate the issue of multiple passes over the same video frames. However, all of these approaches require manual annotations for atomic actions which is subjective, laborious, and time-consuming. Shou et al. [34] proposed a multi-stage CNN to solve the temporal localization problem. Although it relaxes the requirement of exact temporal annotation, its benefit is overshadowed by the complexity of multi-scale candidate segment generation and multiple network training.

Approaches that completely forgo action-level temporal annotation generally use a learning framework for multiple instance selection [25] [26] [37]. This allows localization of action or anomalous events by finding key instances in untrimmed videos. The video segments are considered instances, and the key instances are learned based on only video-level labels.

Object Detection: Object detection refers to identifying an object and its localization. Region-Based Convolutional Neural Networks (R-CNN) [11] have shown impressive results in object detection. The process includes a sequence of CNN-based feature extraction, object classification, and bounding box regression. Mask R-CNN generates a mask in pixel level of the object to segment it from the generated proposals [12]. Faster-RCNN uses a dedicated CNN-based Region Proposal Network (RPN) that drastically reduces the proposal generation time [32].

Injury Detection: Injury detection or prediction is a well-studied area in sports analytics. However, most research, especially for American football, are based on either physical and psychological statistics of the player [7] [20] or data collected from micro-sensor [19] [42] and manual investigation of incident videos [28] [39] [19]. Very recently, [31] successfully applied 3D convolution to early detection of injury in baseball pitchers using only videos. There is hardly any video-based work for American football that attempts to identify risky tackles that may result in serious head injury.

Table 1: Data distribution in the data set

Class Name	No. of Samples	Avg No. of Frames
Safe	123	216
Risky	55	203
Total	178	212

3 Data Set Preparation

Lack of a data set relevant to our task motivates us to construct a new data set. We attempt to solve the problem from a supervised learning point of view; therefore, we need labeled training data. We build our data set in two steps. First, we collect videos from practice fields, and then we label each video manually.

3.1 Video Collection

Our data set consists of 178 tackle videos. Originally, we collected other videos as well, but we had to discard some because of poor resolution and older encoding format. All the videos are collected from seven different practice fields in the United States. They are recorded in different formats: MOV, MOD, MKV, and MP4. All files are then converted to MP4. The frame rate for all videos is 30. A standard guideline was used to set up cameras, however, the guideline was not strictly maintained. In all videos, the player starts running from the left, and the dummy is placed on the right. Some unnecessarily long videos are trimmed to some extent.

3.2 Data Annotation

We consider the task of risky tackle identification as a binary classification problem. Therefore, we annotate each video as either 'safe' or 'risky'. The annotation is done by a certified athletic trainer who first rates every tackle on a scale of 3. Tackles scored 0 or 1 are considered risky while tackles scored 2 and 3 are considered safe. The annotator judged every video based on the head position, body posture, and contact point around the strike zone: where the player hits the dummy. Although many factors from consecutive frames are involved, loosely speaking, if the head or helmet of the player initiates the contact, the tackle is risky, but if the player uses his chest or shoulder for initial contact keeping his head away, that is considered a safe tackle. Table 1 shows the data distribution that we have after the preprocessing and annotation.

4 Approach

The main motivation behind our approach is to first extract the spatiotemporal regions that are more relevant to the task with minimum effort and then use

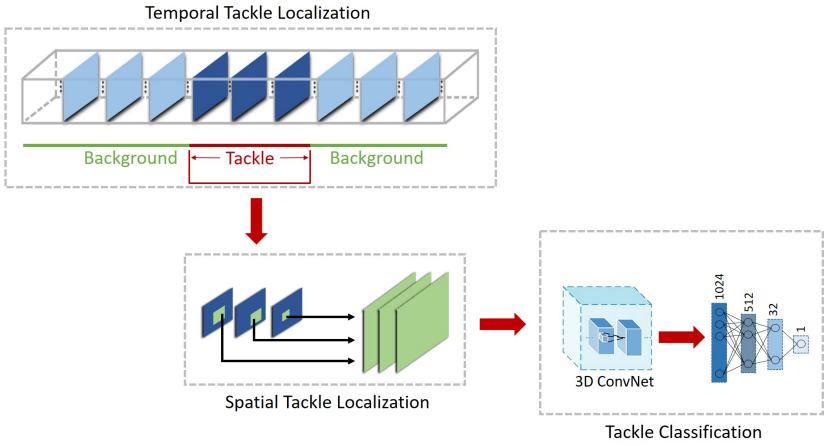


Fig. 2: Overall pipeline. In the first stage of the pipeline, tackle related frames are extracted. The second stage localizes the tackle spatially using a pre-trained Mask-RCNN model. In the final stage, spatiotemporally segmented frames pass through a 3D convolutional network and subsequent fully connected layers.

these informative segments to identify risky tackles. Figure 2 depicts the overall pipeline of our proposed approach.

4.1 Temporal Tackle Localization

Manual investigation reveals that only a few frames around the strike zone contain key information rather than frames that are more distant from the actual tackle event. More specifically, it turns out that only 10-20 frames are important where the tackle is happening compared to the huge number of frames in each video. The task of extracting action-related frames, in other words, the task of removing redundant frames, is similar to action localization. However, the drawback of considering the problem as action localization is that we need the temporal anchors or frame-level annotations defining the start and end of the tackle action in the video. Therefore, general action localization approaches require much effort for annotation. Moreover, such approaches are often multi-stage, which in turn will increase the complexity of our task.

We propose to cast the temporal tackle localization task as anomaly localization where we consider the tackle or collision with the dummy (both safe and risky) as an anomalous event. To avoid the necessity of temporal or frame-level annotations, we leverage a state-of-the-art approach [37] that uses only video-level annotation. They consider each video as a bag and video segments as instances in a deep multiple instance learning framework. To utilize such an approach, we create an auxiliary data set. From one video of the original data set, we create two videos, one before the tackle occurs and another after the

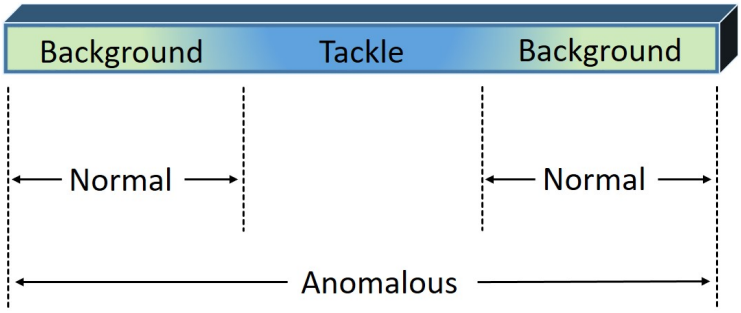


Fig. 3: Visualization of auxiliary data set creation. Our approach does not consider the highly specific start or end point of the tackle event. Any point in the gradient space avoiding the solid blue zone can be considered as the temporal anchor for the normal video.

tackle is finished, unless the tackle event is at the very start or end of the video. However, this has been done without considering the precise temporal location of the tackle, as shown in Figure 3. It just requires that the normal videos will not contain the core frames of the tackle event; thus, anyone with no domain knowledge can perform the task of video clipping. We consider these newly created videos as normal videos because they do not contain any tackle where the player is hitting the dummy. On the other hand, we use the original untrimmed videos as anomalous videos because they contain either safe or risky tackle events at some point in the video. The rationale behind using the untrimmed videos as anomalous videos instead of video of the clipped tackle event is twofold. First, the frames that do not contain the tackle event reside in both normal and anomalous videos. As [37] uses a ranking loss function that discriminates the highest scored instances in the normal and anomalous videos, the presence of non-tackle segments in both normal and anomalous videos inherently increase the score of the tackle segment. Second, at the test time, we expect our approach to identify the tackle related frames from the untrimmed videos. Thus, training the model using the whole videos resonate better with the end goal.

We train the anomaly detection model with these normal and anomalous videos, so the model learns to predict anomaly scores for each segment of a video. We propose an anomaly score-based selection mechanism for frame extraction. We run a temporal sliding window of 16 frames with 8 frame overlap and predict the anomaly score for each window. First, we select the window i with the highest anomaly score. Then the window which scores higher between the two window $i+1$ and $i-1$ is selected. Finally, we take an extra four frames before and after the two selected consecutive windows. In this way, the 32-frame long window or segment will contain the anomaly: the tackle event.

4.2 Spatial Tackle Localization

The tackle event takes place only in a particular spatial region of a frame within a large background. Figure 1 clearly shows a background containing unnecessary information such as other players, coaches, playground infrastructure, and service cart. Thus, considering only the spatial region of interest can drastically reduce the spatial dimension without the loss of any key information.

We propose to take the advantage of a pre-trained object recognition model to spatially localize the tackle event. As the tackle is performed by a person, we use the Mask R-CNN [12] object detection model to generate the bounding boxes for all persons present in the frame. The player appears in a wide bounding box because of the action performed and the camera set up in close proximity. Thus, we exploit the relative width of the bounding boxes to select the player performing the tackle when several persons are present within a frame. Finally, the selected bounding box is extended to the top and right sides to include the dummy.

4.3 Tackle Classification

We utilize a 3D convolutional network to learn the spatiotemporal features from the video frames we retain after discarding the redundant frames. Specifically, our architecture includes four 3D convolution layers each followed by a 3D max-pooling layer. The number of filters for the four convolution layers are 16, 64, 256, and 1024, respectively. We use $3\times3\times3$ convolution filters with stride $1\times1\times1$ for all layers. According to [40], to preserve temporal features in the first stage, we have a kernel size of $1\times2\times2$ and stride $1\times2\times2$ for the first pooling layer. All other 3D pooling layers are $2\times2\times2$ with stride $2\times2\times2$. A global average pooling layer connects the 3-layer fully connected (FC) block to the convolutional block. The first FC layer has 512 units, the second layer has 32 units, and the final layer has only 1 unit. ReLU activation is used for all the layers except the final one, which has Sigmoid activation. We apply 50% dropout regularization after each FC layer and use Adam optimizer with a learning rate of 0.0001. We perform parameter sweep to empirically select the best set of hyperparameters for the network. The model is trained to minimize the binary cross-entropy loss:

$$\mathcal{L} = \sum_i (y_i \; log \, p_i + (1 - y_i) \; log \, (1 - p_i)), \tag{1}$$

where y_i and p_i denotes the label and the prediction, respectively, for sample i.

5 Experimental Setup

5.1 Train-Test Split

Experiments were repeated three times with random splits of the data. In each trial, we perform a 80%-20% train-test split over the samples in the data set. In all splits, we maintain approximately the same distribution of classes as in the original data set. To ensure a robust generalizable analysis, we hold out the test set at all stages of the pipeline.

5.2 Baseline

We compare our method with one state-of-the-art video classification and one anomaly detection approach to evaluate the effectiveness of our proposed pipeline.

C3D Baseline [40]: We follow the same convention as mentioned in [40] to extract the C3D descriptor from the whole video. We obtain the fully connected (FC) layer FC6 activations of the C3D network for each 16 frame clip with an eight frame overlap. Finally, to get the C3D video descriptor, we average these clip level activations and then apply l_2 normalization that results in a 4096-dim vector. At first, we attempt to learn a Support Vector Machine (SVM) classifier using the C3D video descriptor as originally used in [40]. However, such shallow models fail to learn anything meaningful and always tend to predict the majority class. This may be due to the class imbalance and lack of sufficient representative samples from the minority class. The use of class weightage does not seem to help. Thus, we use a Multi Layer Perceptron (MLP) similar to the one described earlier in Section 4.3 as the classifier.

Anomaly Detection (AD) Baseline [37]: We compare our model with this state-of-the-art approach because the task of risky tackle detection can be considered as an anomaly detection task. We train their network assuming the risky tackles as anomalous events and the safe tackles as normal events. We use exactly the same settings for the hyperparameters as in the original implementation.

5.3 Evaluation Metrics

We consider the set of risky tackles as the positive class while safe tackles are the negative class. In the presence of class imbalance, which can be extreme for practice tackles that are supervised by coaching staff, accuracy cannot serve as an adequate figure of merit, because it does not consider the skewness in class distribution. Therefore, we report balanced accuracy, which is defined as

$$\frac{True\ Positive\ Rate + True\ Negative\ Rate}{2}.$$

Following earlier works on learning with imbalanced data sets, we also report precision, recall, and F1-score to compare the performance of risky tackle identification at the final stage. Further, we qualitatively evaluate the performance of the intermediate stages.

6 Results and Discussions

Comparison with the Baseline: Table 2 shows the quantitative experimental results for both the baselines and our proposed approach. Under the name Temporal Localization (TL) only, we report the classification results using the frames obtained just after the first stage of the pipeline. This also serve as an ablation study for the temporal localization stage. When the Spatial Localization (SL) is added on top of temporal localization, we use TL + SL to refer to it.

Table 2: Evaluation metrics for different approaches (TL: Temporal Localization, SP: Spatial Localization)

Methods	Bal. Acc.	Precision	Recall	F1-Score
C3D (untrimmed)	54.81	41.67	22.22	28.05
AD (untrimmed)	53.53	35.71	41.67	38.46
Ours (TL Only)	55.13	47.62	33.33	37.61
Ours (TL + SL)	**66.88**	**54.25**	**55.56**	**54.29**

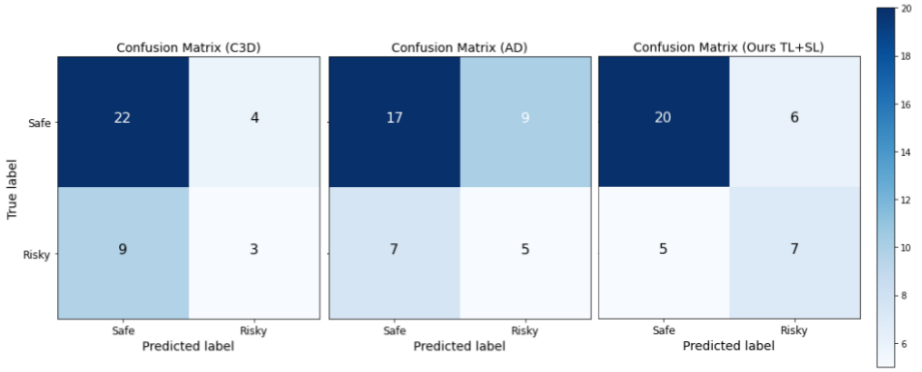


Fig. 4: Confusion matrices of a particular trial for C3D baseline (left), state-of-the-art anomaly detection model (middle), and our proposed approach (right).

Table 2 shows that our proposed 3-stage pipeline outperforms the C3D and Anomaly Detection (AD) based approaches applied to the untrimmed videos in terms of all metrics. Our approach achieves $12 - 13\%$ higher balanced accuracy than the other approaches, which means it detects both classes better. In particular, our 2-stage (TL only) approach achieves 6% higher precision than the C3D baseline and and 12% higher precision than the AD baseline. It further improves by 6% when combined with spatial localization. In terms of recall, our TL only approach does not do better than the AD baseline, however, the 3-stage (TL+SL) approach achieves significantly higher recall than C3D and outperforms the AD baseline by a considerable margin. Moreover, our (TL+SL) approach is able to achieve an F1-score of 54.29, which is 26% and 15% better than the C3D and AD baselines, respectively. Therefore, our (TL+SL) approach achieves higher recall and F1-score compared to the other approaches without compromising precision. This denotes the effectiveness of our approach in detecting the positive risky tackle class and maintaining a good balance between positive and negative class detection. Figure 4 presents the confusion matrices for the test set of a particular trial. Our model shows similar time complexity compared to the AD baseline, however, slightly higher than the C3D baseline. The main computational bottleneck stems from the temporal localization stage.

Table 3: Ablation study for temporal and spatial localization

Methods	Precision	Recall	F1-Score
C3D (untrimmed)	41.67	22.22	28.05
C3D (TL Only)	42.86	25.00	30.90
C3D (TL + SL)	52.22	16.67	25.07
Ours (MT Only)	**60.32**	27.22	36.96
Ours (MT + SL)	49.02	30.56	31.49
Ours (TL Only)	47.62	33.33	37.61
Ours (TL + SL)	54.25	**55.56**	**54.29**

Ablation Study: We perform an ablation study to analyze the necessity and efficacy of the intermediate stages. We also manually localize the tackle for robust comparison and report the results using the notation Manually Trimmed (MT). We answer the following research questions to interpret the experimental results:

– *Is there any improvement in performance due to the temporal localization?*

As we can see from Table 3, the TL and MT only approaches always outperform the C3D baseline for untrimmed video in all aspects. We have extracted 32 frames while the average number of frames in untrimmed videos is 212. Thus, removing a significant portion of the frames does not affect the performance negatively rather improves it and saves computation power. Also, this reduction in the number of frames opens up the possibility of learning spatiotemporal features directly from all frames instead of averaging the features obtained from chunked video clips.

– *Does the use of the spatial localization improve the classification performance?*

Table 3 shows that the addition of the spatial localization stage always increases the recall for our proposed model compared to the TL or MT only counterparts. That means removing unnecessary spatial information contributes largely to improve the detection of risky tackles. However, the C3D model performs poorly when combined with spatial localization, possibly because of the biases of the pre-trained C3D features towards unsegmented frames.

Accuracy of Temporal Localization: We manually evaluate whether tackles are present in the 32-frame long clips extracted from all the test videos. It turns out that the localization model trained in stage one achieves 97% accuracy in the test set for the temporal localization task. Therefore, we conclude subsequent stages will require more attention to improve the overall classification performance.

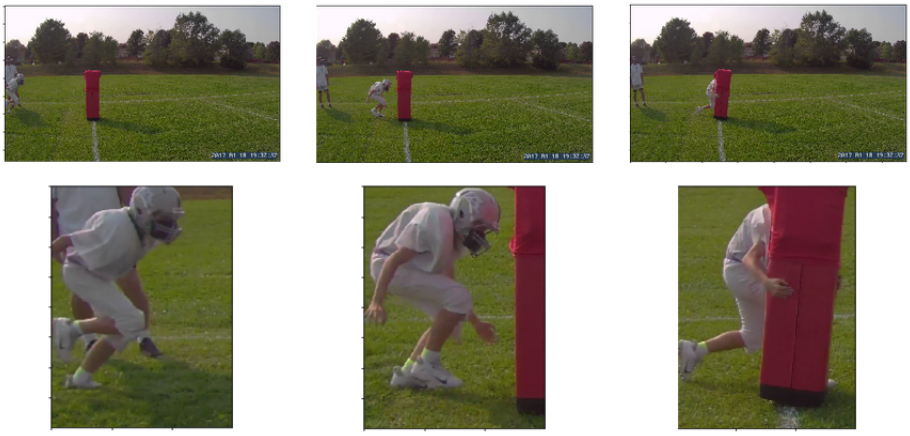


Fig. 5: Example of spatial localization. The top row shows the original frames and the bottom row presents the corresponding spatially segmented ones.

Accuracy of Spatial Localization: We leveraged the Mask-RCNN model from the Detectron2 [43] library in such a way that the detection of the player is guaranteed in most cases. Figure 5 presents some examples of spatial localization. However, when the player has not entered the camera's field of view and there is another person in the frame, our method occasionally selects that person as a player. Also, just after the tackle when the player trends downward with the dummy, the player is often occluded by the dummy. This may cause the spatial localization model to fail. However, such failures may not affect the detection task significantly, because these scenarios arise either before the tackle event has started or after hitting the dummy.

7 Conclusions

In this paper, we propose a 3-stage pipeline to detect risky tackles from American football practice videos. The experimental results show that our proposed method performs significantly better than the existing 3D ConvNet-based methods for video classification. There are limitations due to the size of our presented data set, however, the inherent skewness poses a substantial challenge for improving the model performance even beyond the random guess. In the future, we would like to use multi-modal approaches to take the benefit of optical flow features and pose estimation. The use of this automatic risky tackle identification framework can provide faster feedback to the player, and such feedback and supervision during the tackle practice can significantly minimize the risks of head impact and head injuries among young American football players.

Acknowledgements. We would like to thank Ademola Okerinde for his insightful discussions and Nazmun Akter Pia for her assistance in creating the visualizations.

References

1. Alosco, M., Kasimis, A., Stamm, J., Chua, A., Baugh, C., Daneshvar, D., Robbins, C., Mariani, M., Hayden, J., Conneely, S., et al.: Age of first exposure to american football and long-term neuropsychiatric and cognitive outcomes. Translational psychiatry **7**(9), e1236–e1236 (2017)
2. Buch, S., Escorcia, V., Shen, C., Ghanem, B., Carlos Niebles, J.: Sst: Single-stream temporal action proposals. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, pp. 2911–2920 (2017)
3. Carreira, J., Zisserman, A.: Quo vadis, action recognition? a new model and the kinetics dataset. In: proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 6299–6308 (2017)
4. Chrisman, S.P., Lowry, S., Herring, S.A., Kroshus, E., Hoopes, T.R., Higgins, S.K., Rivara, F.P.: Concussion incidence, duration, and return to school and sport in 5- to 14-year-old american football athletes. The Journal of pediatrics **207**, 176–184 (2019)
5. Cournoyer, J., Tripp, B.L.: Concussion knowledge in high school football players. Journal of athletic training **49**(5), 654–658 (2014)
6. Cui, X., Liu, Q., Gao, M., Metaxas, D.N.: Abnormal detection using interaction energy potentials. In: CVPR 2011, pp. 3161–3167. IEEE (2011)
7. Dompier, T.P., Kerr, Z.Y., Marshall, S.W., Hainline, B., Snook, E.M., Hayden, R., Simon, J.E.: Incidence of concussion during practice and games in youth, high school, and collegiate american football players. JAMA pediatrics **169**(7), 659–665 (2015)
8. Donahue, J., Anne Hendricks, L., Guadarrama, S., Rohrbach, M., Venugopalan, S., Saenko, K., Darrell, T.: Long-term recurrent convolutional networks for visual recognition and description. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 2625–2634 (2015)
9. Escorcia, V., Heilbron, F.C., Niebles, J.C., Ghanem, B.: Daps: Deep action proposals for action understanding. In: European Conference on Computer Vision, pp. 768–784. Springer (2016)
10. Giancola, S., Amine, M., Dghaily, T., Ghanem, B.: Soccernet: A scalable dataset for action spotting in soccer videos. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pp. 1711–1721 (2018)
11. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 580–587 (2014)
12. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. In: Proceedings of the IEEE international conference on computer vision, pp. 2961–2969 (2017)
13. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770–778 (2016)
14. Heilbron, F.C., Niebles, J.C., Ghanem, B.: Fast temporal activity proposals for efficient detection of human actions in untrimmed videos. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1914–1923 (2016)

15. Hospedales, T., Gong, S., Xiang, T.: A markov clustering topic model for mining behaviour in video. In: 2009 IEEE 12th International Conference on Computer Vision, pp. 1165–1172. IEEE (2009)
16. Ji, S., Xu, W., Yang, M., Yu, K.: 3d convolutional neural networks for human action recognition. IEEE transactions on pattern analysis and machine intelligence **35**(1), 221–231 (2012)
17. Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., Fei-Fei, L.: Large-scale video classification with convolutional neural networks. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, pp. 1725–1732 (2014)
18. Kay, W., Carreira, J., Simonyan, K., Zhang, B., Hillier, C., Vijayanarasimhan, S., Viola, F., Green, T., Back, T., Natsev, P., et al.: The kinetics human action video dataset. arXiv preprint arXiv:1705.06950 (2017)
19. Kelley, M., Urban, J., Jones, D., Powers, A., Whitlow, C.T., Maldjian, J., Stitzel, J.: Football concussion case series using biomechanical and video analysis. Neurology **91**(23 Supplement 1), S2–S2 (2018)
20. Kelley, M.E., Jones, D.A., Espeland, M.A., Rosenberg, M.L., Miles, C.M., Whitlow, C.T., Maldjian, J.A., Stitzel, J.D., Urban, J.E.: Physical performance measures correlate with head impact exposure in youth football. Medicine and science in sports and exercise **52**(2), 449 (2020)
21. Kelley, M.E., Kane, J.M., Espeland, M.A., Miller, L.E., Powers, A.K., Stitzel, J.D., Urban, J.E.: Head impact exposure measured in a single youth football team during practice drills. Journal of Neurosurgery: Pediatrics **20**(5), 489–497 (2017)
22. Kerr, Z.Y., Roos, K.G., Djoko, A., Dalton, S.L., Broglio, S.P., Marshall, S.W., Dompier, T.P.: Epidemiologic measures for quantifying the incidence of concussion in national collegiate athletic association sports. Journal of athletic training **52**(3), 167–174 (2017)
23. Kerr, Z.Y., Yeargin, S., Valovich McLeod, T.C., Nittoli, V.C., Mensch, J., Dodge, T., Hayden, R., Dompier, T.P.: Comprehensive coach education and practice contact restriction guidelines result in lower injury rates in youth american football. Orthopaedic journal of sports medicine **3**(7), 2325967115594,578 (2015)
24. Klaser, A., Marszałek, M., Schmid, C.: A spatio-temporal descriptor based on 3d-gradients. In: BMVC 2008-19th British Machine Vision Conference, pp. 275–1. British Machine Vision Association (2008)
25. Lai, K.T., Liu, D., Chen, M.S., Chang, S.F.: Recognizing complex events in videos by learning key static-dynamic evidences. In: European Conference on Computer Vision, pp. 675–688. Springer (2014)
26. Lai, K.T., Yu, F.X., Chen, M.S., Chang, S.F.: Video event detection by inferring temporal instance labels. In: Proceedings of the ieee conference on computer vision and pattern recognition, pp. 2243–2250 (2014)
27. Laptev, I.: On space-time interest points. International journal of computer vision **64**(2-3), 107–123 (2005)
28. Lessley, D.J., Kent, R.W., Funk, J.R., Sherwood, C.P., Cormier, J.M., Crandall, J.R., Arbogast, K.B., Myers, B.S.: Video analysis of reported concussion events in the national football league during the 2015-2016 and 2016-2017 seasons. The American journal of sports medicine **46**(14), 3502–3510 (2018)
29. McAllister, T., McCrea, M.: Long-term cognitive and neuropsychiatric consequences of repetitive concussion and head-impact exposure. Journal of athletic training **52**(3), 309–317 (2017)

30. Oneata, D., Verbeek, J., Schmid, C.: Action and event recognition with fisher vectors on a compact feature set. In: Proceedings of the IEEE international conference on computer vision, pp. 1817–1824 (2013)
31. Piergiovanni, A., Ryoo, M.S.: Early detection of injuries in mlb pitchers from video. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, pp. 0–0 (2019)
32. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. arXiv preprint arXiv:1506.01497 (2015)
33. Scovanner, P., Ali, S., Shah, M.: A 3-dimensional sift descriptor and its application to action recognition. In: Proceedings of the 15th ACM international conference on Multimedia, pp. 357–360 (2007)
34. Shou, Z., Wang, D., Chang, S.F.: Temporal action localization in untrimmed videos via multi-stage cnns. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1049–1058 (2016)
35. Simonyan, K., Zisserman, A.: Two-stream convolutional networks for action recognition in videos. arXiv preprint arXiv:1406.2199 (2014)
36. Smart, B.J., Haring, R.S., Asemota, A.O., Scott, J.W., Canner, J.K., Nejim, B.J., George, B.P., Alsulaim, H., Kirsch, T.D., Schneider, E.B.: Tackling causes and costs of ed presentation for american football injuries: a population-level study. The American journal of emergency medicine **34**(7), 1198–1204 (2016)
37. Sultani, W., Chen, C., Shah, M.: Real-world anomaly detection in surveillance videos. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 6479–6488 (2018)
38. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 2818–2826 (2016)
39. Tierney, G.J., Kuo, C., Wu, L., Weaving, D., Camarillo, D.: Analysis of head acceleration events in collegiate-level american football: A combination of qualitative video analysis and in-vivo head kinematic measurement. Journal of Biomechanics **110**, 109,969 (2020)
40. Tran, D., Bourdev, L., Fergus, R., Torresani, L., Paluri, M.: Learning spatiotemporal features with 3d convolutional networks. In: Proceedings of the IEEE international conference on computer vision, pp. 4489–4497 (2015)
41. Wang, H., Schmid, C.: Action recognition with improved trajectories. In: Proceedings of the IEEE international conference on computer vision, pp. 3551–3558 (2013)
42. Wu, L.C., Kuo, C., Loza, J., Kurt, M., Laksari, K., Yanez, L.Z., Senif, D., Anderson, S.C., Miller, L.E., Urban, J.E., et al.: Detection of american football head impacts using biomechanical features and support vector machine classification. Scientific reports **8**(1), 1–14 (2017)
43. Wu, Y., Kirillov, A., Massa, F., Lo, W.Y., Girshick, R.: Detectron2. https://github.com/facebookresearch/detectron2 (2019)
44. Xu, D., Ricci, E., Yan, Y., Song, J., Sebe, N.: Learning deep representations of appearance and motion for anomalous event detection. arXiv preprint arXiv:1510.01553 (2015)
45. Zhao, B., Fei-Fei, L., Xing, E.P.: Online detection of unusual events in videos via dynamic sparse coding. In: CVPR 2011, pp. 3313–3320. IEEE (2011)

Analysis of the Behavior of Online Decision Trees Under Concept Drift at the Example of FIMT-DD

Marcel Hanitz, Marvin Schöne, Tim Voigt, and Martin Kohlhase

Center for Applied Data Science Gütersloh, Faculty of Engineering and Mathematics, Bielefeld University of Applied Sciences, Germany
{mhanitz,marvin.schoene,tim.voigt,martin.kohlhase}@fh-bielefeld.de

Abstract. In online learning, the detection of concept drift is still a challenging task. To avoid incorrect or missing model adjustments due to concept drift and to increase confidence in the model, model changes must be transparent to the user. This paper presents a novel approach to detect and visualize model changes of decision trees in supervised online learning which are caused by concept drift. The approach is tested on a synthetic data stream containing global abrupt concept drift at two predefined time points. For supervised online learning, the *Fast Incremental Model Tree with Drift Detection* (FIMT-DD) is used. It is shown that the FIMT-DD reacts to concept drift differently, depending on the deterioration of the prediction error. The first concept drift that causes a large increase of the prediction error by the factor of ten leads to approximately 80 % of the tree structure being replaced. The second concept drift with a comparatively smaller degradation of the prediction error results in the replacement of less than 20 % of the tree structure. Replacing subtrees and thus adapting to changes in the data stream leads to an improvement of the prediction error in both cases.

Keywords: concept drift · online learning · decision trees · FIMT-DD.

1 Introduction

Machine learning has been used frequently for the improvement of industrial production processes in recent years. Machine learning can be used, for example, to predict whether and when machines will need repairs (predictive maintenance) [3] or whether products will meet quality specifications [15].

The predictions are made based on models learned from data collected in the past. When functional relationships in data change, e.g. due to variations in a manufacturing process, the prediction accuracy of a model trained with historical data often deteriorates. To maintain a certain prediction accuracy, models have to be retrained manually. So-called online learning algorithms that counteract this have become popular in recent years [20]. These algorithms are not trained with a previously collected data set, but with a continuous data stream. In this way, the model is automatically adjusted to new patterns in the

data, maintaining prediction accuracy and avoiding manual retraining. In the literature, there are no consistent definitions of the term *online learning*. Often the definitions provided include the properties adaptivity, incremental model adaptation and constant memory usage [8]. In [29] online learning is defined as follows: An online learning procedure generates on the basis of a data stream $\mathcal{S}(k) = \{s(1), s(2), \ldots, s(k)\}$ with the data points $s(1), s(2), \ldots, s(k)$ a sequence of models $h(1), h(2)$, ..., $h(k)$. Here, k denotes to the index of the current data point from the data stream. Each data point $s(k) = (\mathbf{x}^{\mathrm{T}}(k), y(k))$ consists of a feature vector $\mathbf{x}^{\mathrm{T}}(k) = [x_1(k)\ x_2(k) \ldots x_p(k)]$, containing the values of the p features x_1 to x_p at time point k, and the corresponding target value $y(k)$.

Each feature vector $\mathbf{x}^{\mathrm{T}}(k)$ and target value $y(k)$ can be described as a realization of a random variable X and Y respectively, with Y depending on X. Changes over time of a process, which affect the joint probability density function (pdf) of the data distribution $P(X, Y) = P(Y|X) \cdot P(X)$, are called concept drift [14]. There are three types of changes: The pdf $P(X)$ of the feature vector can change, leading to data points in previously excluded regions of the input space (virtual concept drift), the conditional pdf $P(Y|X)$ can change due to non-stationary process behavior (real concept drift), both $P(X)$ and $P(Y|X)$ can change [29]. Real concept drift can occur gradually, abruptly, incrementally, or recurrently. For example, the wear of a machine is usually gradual, whereas a repair of a machine causes an abrupt change in data.

A disadvantage of online learning algorithms is that they can adapt to changes they are not supposed to adapt to, e.g. due to an incorrectly adjusted sensor. Therefore, it is relevant that the user of the online algorithm understands how the model changes over time. To make changes of the model understandable, two requirements must be fulfilled. First, both the behavior and the structure of the model must be interpretable. Second, online adjustments of the model must be visualized. Due to their inherent interpretability, decision trees are well suited to satisfy the first requirement, which has already been proven in numerous offline applications [25]. However, there are also promising methods for online learning of decision trees, like the *Fast Incremental Model Trees with Drift Detection* (FIMT-DD) [22]. Similar to the first requirement, methods to visualize model changes are primarily available for offline learning, so the second requirement is still pointed out as a topic of research in online learning [12, 17].

In this paper, a novel approach to visualize model changes of decision trees during supervised online learning is presented. The approach is investigated using FIMT-DD and abrupt concept drift as case studies, but is not limited to them. To perform an extensive study, FIMT-DD is trained with a synthetic data stream to which concept drift is added at predefined points in time. Before and after the occurrence of abrupt concept drift, the tree structure and prediction accuracy of FIMT-DD are examined and visualized in two-dimensional subspaces of the input space where the model changes occur.

The paper is organized as follows: Section 2 discusses related work in online learning and Section 3 explains the basics of the FIMT-DD algorithm. Section 4

deals with the experimental analysis of the FIMT-DD in the context of concept drift. Finally, Section 5 summarizes the results of this paper.

2 Related Work

Depending on the availability of the target value, online learning can be grouped into supervised, semi-supervised and unsupervised online learning [20]. In contrast to semi-supervised and unsupervised online learning, supervised online learning requires that a target value is available at each incremental learning step k that can be assigned to the feature vector such that the data point $s(k) = (\mathbf{x}^{\mathrm{T}}(k), y(k))$ is given.

The application of supervised online learning extends over many disciplines. In [24], different online regression algorithms are tested on synthetic and real-world data for exoskeleton control. The application in [26] aims to predict short-term traffic flow using *Online Learning Weighted Support-Vector Regression*. Another real-world application can be found in [16], where *Probabilistic Neural Networks* are used for a dynamic security classification of electric power systems. In [32], *Radial Basis Function Networks* are used for both classification (detection of handwritten digits) and regression (time-series prediction). A slightly different approach is presented in [21], where a combination of different models are continuously adapted by online batch learning to predict the product quality in a semiconductor assembly process.

In addition to the already named online learning methods, an online support vector machine is presented in [36] and an extensive study of supervised online learning with different neural networks is carried out in [23]. Motivated by well established tree construction algorithms and ensemble methods for offline learning like CART [2], GUIDE [28] or Random Forest [1], many tree-based online learning methods have been developed apart from FIMT-DD: *The Very Fast Decision Tree* [7] for classification and its extensions [11, 30], the *Online Random Forest* [35] and *Ultra Fast Forest Tree system* [10] for classification, and the *Mondrian Forest* [27] for regression. In [37], a supervised dimension reduction method called *Incremental Sliced Inverse Regression* is described that produces a linear regression model. The *Passive-Aggressive Algorithm* of [6] is applicable for both classification and regression, using a margin based approach which only adjusts the model to data points that were predicted insufficiently. The retraining of the regression model is similar to *Recursive Least Squares* [34].

An extensive study in [29] has shown that most supervised online learning algorithms cannot handle concept drift, or can only handle it to a limited extent. FIMT-DD solves this problem by using a specific approach to detect concept drift, which is further described in Subsection 3.3. The methods in [4] and [18] use a semi-supervised and unsupervised approach, respectively, to address this problem. Another challenge is to make model changes transparent to the user that are (supposedly) caused by concept drift, which is discussed in detail in [12, 17]. The user must be able to trust the model to respond to concept drift, and understand how these changes were detected and how the model would adapt.

Moreover, incorporating expert knowledge could improve the handling of concept drift, but locating and explaining changes in the model, which is necessary for this, is described as a major challenge.

3 FIMT-DD-Algorithm

Several online regression methods were proposed that are suited for dealing with data streams [5]. In this paper, the FIMT-DD algorithm is exemplary regarded, which is used to train online regression trees [22]. Besides the ability to learn from stationary data streams, FIMT-DD contains several mechanisms for detecting and adaption to changes (concept drift) occurring in non-stationary data streams.

A brief overview of the online training process of FIMT-DD is given in Fig. 1. All presented steps are performed in each iteration of the learning process. Beginning with just one leaf, the tree structure is expanded by further splitting the leaves. In order to adapt to concept drift, the FIMT-DD can replace outdated nodes and leaves. In the following the most important elements of FIMT-DD are described.

3.1 Splitting Criterion

The decision whether and where to split a leaf is essential for incorporating new information into the model structure without overfitting. In FIMT-DD, this decision is statistically made, based on the Hoeffding bound [19].

For splitting a leaf, a combination of a feature x_i with $i \in \{\mathbb{N} \,|\, 1 \leq i \leq p\}$ and a threshold value $a \in \mathbb{R}$ must be selected, defining the splitting condition $c : x_i \leq a$. FIMT-DD uses the Standard Deviation Reduction (SDR) method to asses the quality of different splits. The SDR is an incremental method, enabling an efficient computation [22]. It is based on the standard deviation of a single leaf j

$$\mathrm{sd}(\mathcal{S}_j) = \sqrt{\frac{1}{N_j}\Big(\sum_{(\mathbf{x}^{\mathrm{T}}(l),y(l))\in\mathcal{S}_j} y(l)^2 - \frac{1}{N_j}\Big(\sum_{(\mathbf{x}^{\mathrm{T}}(l),y(l))\in\mathcal{S}_j} y(l)\Big)^2\Big)}, \tag{1}$$

with $l \in \{\mathbb{N} \,|\, 1 \leq l \leq k\}\}$ and $\mathcal{S}_j \subseteq \mathcal{S}(k)$ denoting a subset of $N_j = |\mathcal{S}_j|$ data points that have reached the leaf. The SDR describes the extent to which the standard deviation of the data set $\mathcal{S}_j$ is reduced when it is split into the data sets $\mathcal{S}_{\mathrm{L}} \subseteq \mathcal{S}_j$ and $\mathcal{S}_{\mathrm{R}} := \mathcal{S}_j \setminus \mathcal{S}_{\mathrm{L}}$. For a possible split in x_i, the SDR is given by

$$\mathrm{SDR}(a) = \mathrm{sd}(\mathcal{S}_j) - \frac{N_{\mathrm{L}}}{N_j}\mathrm{sd}(\mathcal{S}_{\mathrm{L}}) - \frac{N_{\mathrm{R}}}{N_j}\mathrm{sd}(\mathcal{S}_{\mathrm{R}}). \tag{2}$$

The weighted standard deviations of the hypothetical new leaves are subtracted from the standard deviation of the original leaf, with N_{L} and N_{R} denoting the number of data points in the left and right leaves, respectively.

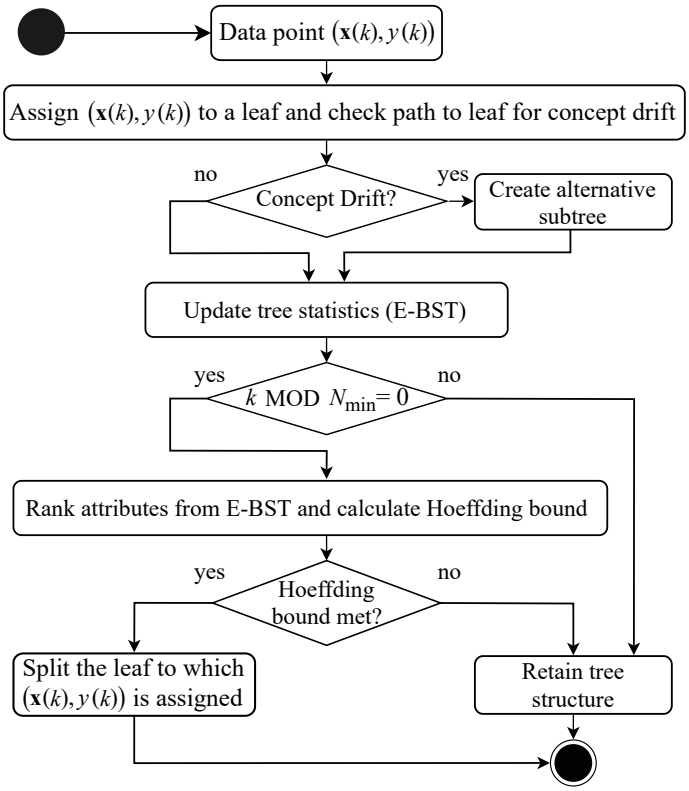


Fig. 1: Flow chart with the most important steps in the learning process of FIMT-DD. Concept drift detection is performed with every new data point, whereas updates to the tree structure are only made each $N_{\min}$ data points.

As can be seen in Equation (1), the calculation of the standard deviation, which is a prerequisite for calculating the SDR, is based on the sum of the target values y, the sum of the squared values of y and the number N_j of training instances that have reached a leaf. FIMT-DD stores all these statistical information needed in form of Extended Binary Tree Structures (E-BTS), with separate E-BTS for each numerical feature x_i. Each node in the E-BTS corresponds to a possible split point and is updated with each data point reaching it [22].

After every $N_{\min}$ data points seen, the best possible split for each feature is selected from its corresponding E-BTS. These p splits are ranked based on the SDR. The split of feature x_i with the highest value is called $c_{1\text{st}}$, whereas the second best split of feature x_n with $n \neq i$ is called $c_{2\text{nd}}$. Now the ratio

$$r_j = \text{SDR}(c_{2\text{nd}})/\text{SDR}(c_{1\text{st}}) \tag{3}$$

between the two SDR values that result of these two splits is regarded. With this ratio the Hoeffding bound can be calculated, which is used to decide whether to

perform a split of the leaf or not. The Hoeffding bound [19] is given by

$$\varepsilon_j = \sqrt{\frac{R^2 \ln(1/\delta)}{2N_j}}. \tag{4}$$

It enables to state with a minimum confidence $1 - \delta$ that the average of the samples with values in the range R lies within distance ε_j of the true mean, i.e. the mean that one would expect from the actual distribution [22]. If $r_j > \varepsilon_j$, a split of the leaf is performed. The main advantage of using the Hoeffding bound is that a decision can be made with statistical certainty on whether to split a leaf or not. In general, the more data points there are in the leaf, the more likely is the split.

3.2 Leaf Models

In order to make predictions, a data point $(\mathbf{x}^{\mathrm{T}}(k), y(k)) \in \mathcal{S}_j$ is passed to the suited leaf j. The local model structure for the prediction of the target value in leaf j is defined as a perceptron

$$\hat{y}_j(k) = w_{j,0} + \sum_{i=1}^{p} w_{j,i}(k) \cdot x_i(k). \tag{5}$$

Each perceptron consists of a bias weight $w_{j,0}$ as well as the weights $w_{j,i}$, depending on the number of input values p. In the learning process, the weights $w_{j,i}$ are updated in an incremental fashion by using the delta-rule. When a new data point is allocated to a leaf, the weights in the perceptron of the leaf

$$w_{j,i}(k) = w_{j,i}(k-1) + \eta(\hat{y}_j(k) - y(k))\tilde{x}_i(k), \quad i \neq 0 \tag{6}$$

are updated. The new weight $w_{j,i}(k)$ depends on the old weight $w_{j,i}(k-1)$, the difference between prediction and target value, multiplied by the learning rate η and the normalized value of the feature $\tilde{x}_i$. For incremental normalization of features, the FIMT-DD uses an adapted version of the studentized residual. The learning rate η can either be kept constant on a small value (e.g. 0.01) or be adapted to the number of seen samples in a leaf [22].

3.3 Concept Drift Detection

Concept drift can degrade the accuracy of a decision tree either locally or globally. If local concept drift occurs in a region of the input space, only parts of the decision tree become worse or unusable. Global concept drift, with changes in the whole input space, affects the whole decision tree. To handle both types of concept drift, FIMT-DD monitors all nodes and leaves for concept drift and, if necessary, replaces affected parts. For this monitoring, a lightweight on-line change detection is used, which is next described in more detail.

Change Detection To monitor for concept drift, the prediction error of each node is assessed. An increase of the prediction error indicates that the tree does not represent the input data as well as before and there may be concept drift. Since the FIMT-DD predictions are made only by using the leaves, the error of the leaves is propagated up to the individual nodes. To decide whether a change in prediction error is significant and thus likely to be concept drift, the Page-Hinkley test (PH test) is used [31]. The PH test is applied to each individual node j, as soon as a new data point is assigned to it. The test utilizes two variables: The cumulative variable $m_j(k)$ and the minimum value $M_j(k) = \min(\{m_j(q), q \in \{\mathbb{N} \,|\, 1 \leq q \leq k\}\})$ of this variable. The variable $m_j(k)$ is calculated by

$$m_j(k) = \sum_{(\mathbf{x}(l),y(l))\in\mathcal{S}_j} \left(|y(l) - \hat{y}_{j,\mathrm{p}}(l)| - \frac{1}{N_j} \sum_{(\mathbf{x}(o),y(o))\in\mathcal{S}_j} \left(|y(o) - \hat{y}_{j,\mathrm{p}}(o)| \right) - \gamma \right) \quad (7)$$

as the sum of the differences between the absolute prediction errors and the mean of the absolute prediction errors with an additional hyperparameter γ. The hyperparameter controls the sensitivity of the change detection, and the prediction $\hat{y}_{j,\mathrm{p}}(l)$ results from the subtree below node j and is determined by propagating the predictions of the leaves up to node j. The PH test $\mathrm{PH}_j(k) = m_j(k) - M_j(k)$ examines the difference between the two regarded variables. For $\mathrm{PH}_j(k) > \lambda$, with a threshold λ, an alarm is triggered, signaling that concept drift is likely present. The threshold λ has to be chosen by the user as a hyperparameter depending on the allowed rate of (false) alarms. Increasing the threshold decreases the number of alarms and vice versa.

Adaptation Strategy The FIMT-DD follows an adaptation strategy where, when the PH test alarms, the affected nodes are first flagged for re-growing. Once a node is appropriately marked, an alternative subtree is created starting in the marked node. This alternative subtree is trained in parallel with the existing FIMT-DD, i.e., the alternative subtree is trained with each data point that reaches the marked node in the original tree. The trees continue to exist in parallel until the alternative subtree becomes better than the original tree in terms of prediction accuracy. If the alternative tree remains worse, a false alarm might have occurred and the alternative tree is removed.

The relative performance between the original subtree $\tilde{h}_{j,\mathrm{o}}$ and the alternative subtree $\tilde{h}_{j,\mathrm{a}}$ is calculated by the so-called Q statistic [13] on the basis of the data points $(\mathbf{x}^{\mathrm{T}}(k), y(k)) \in \mathcal{S}_j$ that have reached the marked node. The squared prediction error is used as a loss function $L(\cdot)$ which is calculated for the original $L(\tilde{h}_{j,\mathrm{o}}, k)$ and alternative subtree $L(\tilde{h}_{j,\mathrm{a}}, k)$. In addition, accumulated sums of these loss functions $S(\tilde{h}_{j,\mathrm{o}}, k-1)$ and $S(\tilde{h}_{j,\mathrm{a}}, k-1)$ up to data point $k-1$ are evaluated for each of these subtrees. To penalize errors further in the past less than recent errors, these accumulated errors are multiplied by a forgetting factor ζ (e.g., $\zeta = 0.995$). The calculation of the Q statistic is given by

$$Q(\tilde{h}_{j,\mathrm{o}}, \tilde{h}_{j,\mathrm{a}}, k) = \log\left(\frac{L(\tilde{h}_{j,\mathrm{o}}, k) + \zeta S(\tilde{h}_{j,\mathrm{o}}, k-1)}{L(\tilde{h}_{j,\mathrm{a}}, k) + \zeta S(\tilde{h}_{j,\mathrm{a}}, k-1)} \right) \quad (8)$$

and is done in a time interval specified by the user, e.g. after every 100 data points. The sign of the value calculated in the Q statistic shows which of the compared trees is more accurate. If the result of the Q statistic is a value greater than zero, the alternative tree performs better than the original tree. Then the alternative tree is inserted into the existing FIMT-DD.

4 Experimental Analysis

The main focus of this paper is to visualize changes in the tree structure of online decision trees caused by concept drift in order to enhance the interpretability and in this way the applicability of these models. This section examines this method at the example of FIMT-DD. In particular, it is shown how FIMT-DD adjusts its model structure in the presence of concept drift and how the adjustments affect the prediction error.

4.1 Experimental Setup

The experiment investigates changes in the prediction error and tree structure of the FIMT-DD when concept drift occurs. The investigation focuses on concept drift that affects the tree structure globally (global concept drift).

Data stream A data stream of training data is passed to the FIMT-DD, which consists in total of 30000 data points. The training and test data are generated using the test function

$$f(x_1, x_2) = 10 \sin(\alpha x_1 x_2) + \tanh\big((x_1 - 0.5)(x_2 - 0.5)\big). \tag{9}$$

This function is an adaption of a function that is used in [9] to evaluate the performance of offline decision trees. The function value $f(x_1, x_2)$ depends on the two input variables x_1 and x_2. By changing the factor α, different types of global concept drift can be artificially generated.

Depending on how the parameter α is changed in the data stream, steady or abrupt concept drift can be generated. Abrupt changes of the factor α are further investigated because in this case the adjustments of the FIMT-DD can be traced more easily. However, due to the functional behavior of the test function, the effect of this global concept drift vary over the input space. The variations of α are visualized in Fig. 2. At certain points of interest (e.g. the points where subtrees are replaced by alternative subtrees), the training with the data stream is interrupted. These points (a)-(f) are referred to as measuring points and are marked in Fig. 2. For each of these measuring points, the current tree structure as well as the prediction accuracy are examined, based on a test data set with $N_{\text{test}} = 10000$ data points.

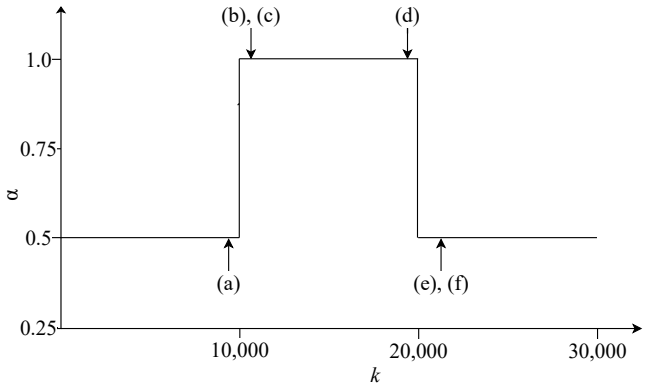


Fig. 2: Variations of parameter α used to create concept drift in the test function. Six different measuring points (a)-(f) are marked, in which the tree structure and the prediction accuracy are examined.

Evaluation criterion: Tree structure In order to make the model structure of decision trees comprehensible for humans (interpretability), the decision boundaries of a decision tree can be plotted [33]. The decision boundaries show which areas of the input space are represented by which leaves. Decision boundaries of an offline decision tree remain static after training until a new training is performed. The decision boundaries online decision trees like the FIMT-DD change depending on the data stream. Thus, leaves can be split further or leaves can be replaced in case of concept drift. In this experiment, the input space is spanned by the two variables x_1 and x_2. Accordingly, it can be visualized in a 2D plot. Each individual leave is represented by a box.

In order to trace the behavior of the FIMT-DD with respect to the adaptation of the tree structure, the points in time when changes in the tree structure occur are investigated. More specifically, two points in time are of special interest: When the FIMT-DD creates new alternative subtrees and when the subtrees are inserted or activated in the FIMT-DD. The time span between these time points describes how quickly an alternative subtree gains a better prediction accuracy than the original subtree. Furthermore, at each measuring point, the number of leaves is considered as well as the level of the decision tree on which the new tree was inserted.

Evaluation criterion: Prediction accuracy The adaptation of the tree structure is intended to ensure that the FIMT-DD represents the current input data well. A good representation can be found when the input data is well generalized (appropriate number of leaves) and the predictions are accurate. In this paper, the accuracy of predictions is measured by the Normalized Root Mean Squared

Error NRMSE. The NRMSE is based on the RMSE,

$$\mathrm{RMSE} = \sqrt{\sum_{n=1}^{N_{\mathrm{test}}} \frac{(y(n) - \hat{y}(n))^2}{N_{\mathrm{test}}}} \tag{10}$$

which describes the squared mean deviation between the predictions $\hat{y}(n)$ and the target values $y(n)$ from the test data set. To calculate the NRMSE, the RMSE is normalized using the difference between maximum $y_{\max}$ and minimum $y_{\min}$ target values from the test data set.

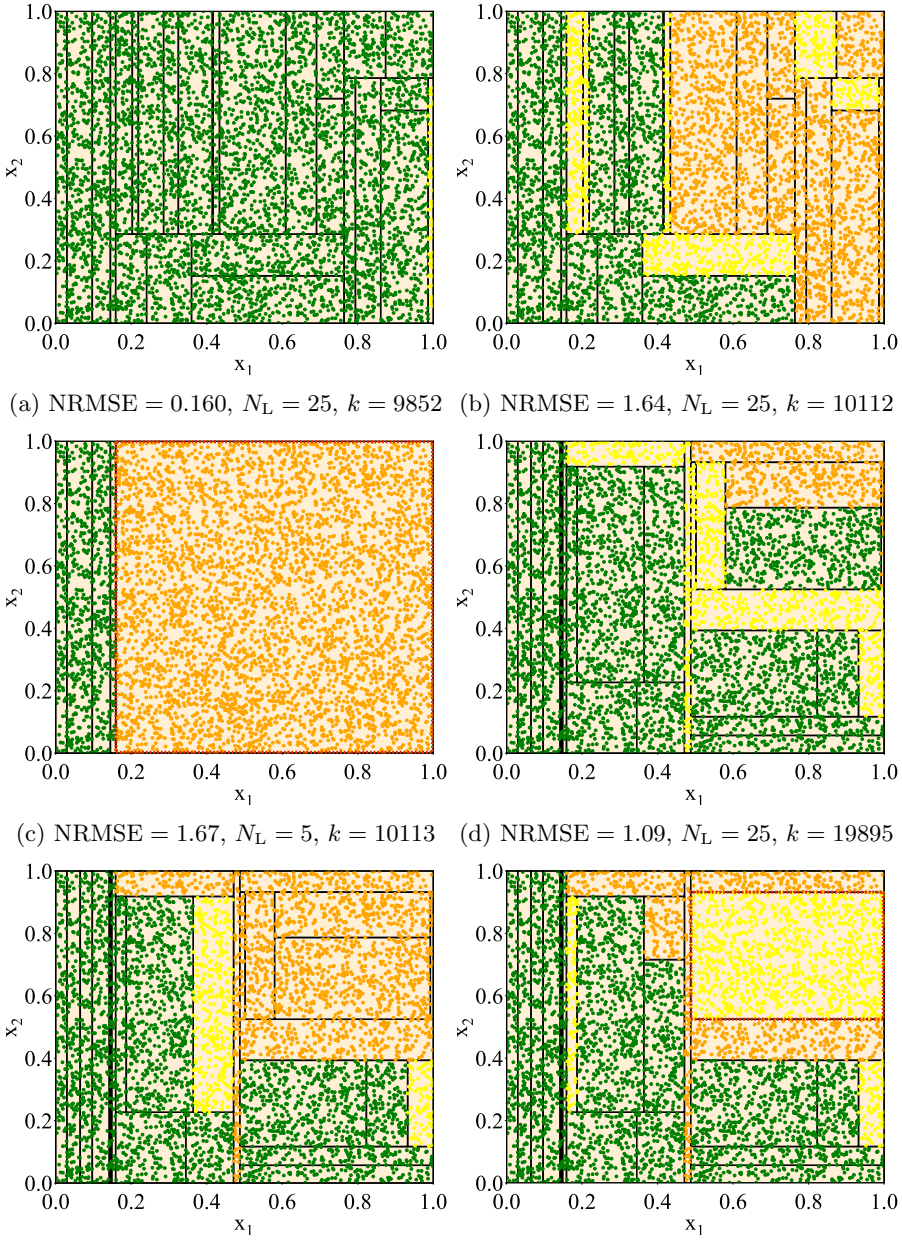
$$\mathrm{NRMSE} = \frac{\mathrm{RMSE}}{y_{\max} - y_{\min}} \tag{11}$$

4.2 Results

This section describes behavior of the FIMT-DD during training with the previously described data stream and changing values for α. Fig. 3 shows the changes of the tree structure and the prediction performance of the FIMT-DD during the experiment. Each subplot shows the tree structure and the prediction performance at one of the measuring points. To ensure good visibility of the tree structure, the number of leaves is limited to a maximum of 25 leaves. The NRMSE is examined both for the tree and separately for individual leaves. For the leaves $\mathrm{NRMSE_L}$ is displayed directly in the plot of the tree structure. For this purpose, each data point from the test data set is plotted in the input space. Depending on the value of $\mathrm{NRMSE_L}$, all data points assigned to the leaf are colored in either green ($\mathrm{NRMSE_L} < 1.2$), yellow ($1.2 \leq \mathrm{NRMSE_L} < 2.4$) or orange ($2.4 \leq \mathrm{NRMSE_L}$), such that green corresponds to a relatively small local prediction error and orange to a relatively high local prediction error. The NRMSE of the whole tree and the number of leaves are indicated below each plot.

Measuring point (a) shows the tree structure after initially learning the test function and before the occurrence of the first concept drift. The NRMSE values of the leaves are in the green range for 24 out of 25 leaves. The NRMSE of the FIMT-DD is 0.160. After 10000 data points, the first concept drift occurs ($\alpha = 1.0$). At measuring point (b) (after 10112 data points), a decrease of the prediction accuracy due to the occurrence of the concept drift is visible. The overall NRMSE increases more than tenfold from 0.16 to 1.64. The errors of the leaves increase especially in the right part of the tree, which can be seen by the orange colored data points in these areas at (b).

The deterioration of the prediction error leads to the tree detecting concept drift. The times at which alternative subtrees are created and the times at which it is inserted into the FIMT-DD are summarized in Tab. 1. Additionally, the hierarchy level at which the concept drift is detected respectively at which the new subtree is supposed to be inserted is given. The first alternative subtree is created at data point 10055 and inserted into the FIMT-DD at data point 10113. It is inserted at the second tree level, such that huge parts of the input space are replaced, as shown at measuring point (c). At this point, a better



(a) NRMSE = 0.160, $N_{\mathrm{L}} = 25$, $k = 9852$ (b) NRMSE = 1.64, $N_{\mathrm{L}} = 25$, $k = 10112$

(c) NRMSE = 1.67, $N_{\mathrm{L}} = 5$, $k = 10113$ (d) NRMSE = 1.09, $N_{\mathrm{L}} = 25$, $k = 19895$

(e) NRMSE = 1.11, $N_{\mathrm{L}} = 25$, $k = 20421$ (f) NRMSE = 1.07, $N_{\mathrm{L}} = 21$, $k = 20422$

Fig. 3: Visualization of the FIMT-DD model structure during training at different measuring points (a)-(f). The local prediction error $\mathrm{NRMSE_L}$ is displayed in color with tree levels: green ($\mathrm{NRMSE_L} < 1.2$), yellow ($1.2 \leq \mathrm{NRMSE_L} < 2.4$) and orange ($2.4 \leq \mathrm{NRMSE_L}$).

Table 1: Points in the data stream of creation and insertion of FIMT-DD subtrees

Subtree	Created (k)	Inserted in FIMT-DD (k)	Level of tree
Subtree 1	10055	10112	2
Subtree 2	10082	-	3
Subtree 3	20159	20421	4
Subtree 4	20161	-	5
Subtree 5	20209	-	4

prediction accuracy at the second tree level can be obtained by using a large submodel rather than by using the 21 outdated (and now removed) subtrees. The left part ($x_1 < 0.17$) of the tree, where only a small degradation of the NRMSE occurs, is retained. Therefore, the tree is able to keep the knowledge which is not slightly affected by the concept drift. Furthermore, after the first concept drift, another subtree was created (at data point 10082), but discarded. Subsequently, the right subtree shown in (c) is further subdivided, leading to the tree shown at $k = 19895$ in (d). The adaption of the FIMT-DD to the training data generated with a $\alpha = 1$ causes a reduction of the NRMSE by about 35 % (from 1.67 to 1.09).

After 20000 data points, the second concept drift occurs, in which the parameter α changes from 1 to 0.5. Due to the change, tree deteriorates in the upper right corner of the input space, as can be seen in (e). The deterioration is less severe than in the first concept drift, leading to a lower number of leaves that are affected by an increasing NRMSE. This can be explained by the fact that the test function is more complex with $\alpha = 1$ and therefore harder to learn than with $\alpha = 0.5$. Three new subtrees are created after the second concept drift, as shown in Tab. 1. However, only one subtree (subtree 3) is inserted into the FIMT-DD. At this point, one subtree is created at the fifth level and two subtrees at the fourth level instead of one subtree at the second level, as it is the case with the first concept drift. This shows that the adaption to concept drift can also happen distributed over several alternative trees. Subtree 3 was created at data point 20159 and inserted into the FIMT-DD at data point 20421. The time between creation and insertion of the subtree is thus more than three times as long as for subtree 1, which was used in the first concept drift. This is due to the fact that the NRMSE does not deteriorate as much in the second concept drift as in the first concept drift. Subtree 3 is inserted at the fourth level of the tree. Therefore, fewer adjustments are required than for the first concept drift, allowing more of the tree structure to be preserved.

5 Summary

This paper presented a novel approach for supervised online learning to monitor when and how decision trees adapt to concept drift. To detect an adaptation of the decision tree, the tree is observed for large structural changes in higher-level nodes of the tree. The model changes are then visualized in two-dimensional sub-

spaces of the input space. In this way, users can understand the online learning behavior, and can detect and avoid an undesirable model adaptation.

The proposed method was tested in an experimental study on a synthetic data stream with abrupt global concept drift at two predefined time points using FIMT-DD. It was shown that the prediction accuracy of certain areas initially deteriorates when concept drift occurs. The number of affected leaves varies depending on how the data stream changes (i.e. if the complexity increases or decreases). FIMT-DD is able to respond appropriately to the changes in the data stream. In the first concept drift, where the NRMSE increases by more than 10 times, a subtree at the second level of the FIMT-DD, and thus a large portion of the tree, is discarded to respond to this high increase in NRMSE. In contrast to the first concept drift, the NRMSE of the tree during the second concept drift increases only slightly and fewer leaves are affected by the concept drift. Here, a subtree is used at the fourth level of the tree. Thus, significantly more information is obtained.

Although the approach has only been tested for regression and an additional offline data set for the visualization, the approach can also be applied to classification and model changes can be visualized online. For the first point, only the error metric must be replaced. For the second point, floating or recursive error measures from the last data points in the data stream can be used.

For further work, the approach should be tested with different types of concept drift (e.g., linear, quadratic) and different tree-based online learning algorithms. Also, the effect of local concept drift can be analyzed. In order to incorporate expert knowledge into the adaptation of the model, the approach can be extended through active learning.

Acknowledgements This work was supported by the EFRE-NRW funding programme "Forschungsinfrastrukturen" (grant no. 34.EFRE-0300180).

References

1. Breiman, L.: Random forest. Machine Learning **45**(1), 5–32 (2001). https://doi.org/10.1023/A:1010933404324
2. Breiman, L., Friedman, J., Stone, C.J., Olshen, R.A.: Classification and Regression Trees. Chapman and Hall/CRC (1984)
3. Carvalho, T.P., Soares, F.A.A.M.N., Vita, R., da P. Francisco, R., Basto, J.P., Alcalá, S.G.S.: A systematic literature review of machine learning methods applied to predictive maintenance. Computers & Industrial Engineering **137**, 106024 (2019). https://doi.org/https://doi.org/10.1016/j.cie.2019.106024
4. Castellani, A., Schmitt, S., Hammer, B.: Task-Sensitive Concept Drift Detector with Constraint Embedding. In: 2021 IEEE Symposium Series on Computational Intelligence (SSCI). IEEE (2021)
5. Choudhary, A., Jha, P., Tiwari, A., Bharill, N.: A brief survey on concept drifted data stream regression. In: Tiwari, A., Ahuja, K., Yadav, A., Bansal, J.C., Deep, K., Nagar, A.K. (eds.) Soft Computing for Problem Solving. pp. 733–744. Springer Singapore, Singapore (2021)

6. Crammer, K., Dekel, O., Keshet, J., Shalev-Shwartz, S., Singer, Y.: Online passive-aggressive algorithms. Journal of Machine Learning Research **7**, 551–585 (2006)
7. Domingos, P.M., Hulten, G.: Mining high-speed data streams. In: KDD '00 (2000)
8. Fontenla-Romero, Ó., Guijarro-Berdiñas, B., Martinez-Rego, D., Pérez-Sánchez, B., Peteiro-Barral, D.: Online machine learning. In: Efficiency and Scalability Methods for Computational Intellect, pp. 27–54. IGI Global (2013). https://doi.org/10.4018/978-1-4666-3942-3.ch002
9. Friedman, J.H.: Multivariate adaptive regression splines. The Annals of Statistics **19**(1), 1–67 (1991)
10. Gama, J.a., Medas, P., Rocha, R.: Forest trees for on-line data. In: Proceedings of the 2004 ACM Symposium on Applied Computing. p. 632–636. SAC '04, Association for Computing Machinery, New York, NY, USA (2004). https://doi.org/10.1145/967900.968033
11. Gama, J.a., Rocha, R., Medas, P.: Accurate decision trees for mining high-speed data streams. In: Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. p. 523–528. KDD '03, Association for Computing Machinery, New York, NY, USA (2003). https://doi.org/10.1145/956750.956813
12. Gama, J.a., Žliobaitundefined, I., Bifet, A., Pechenizkiy, M., Bouchachia, A.: A survey on concept drift adaptation. ACM Comput. Surv. **46**(4) (2014). https://doi.org/10.1145/2523813, https://doi.org/10.1145/2523813
13. Gama, J., Sebastião, R., Rodrigues, P.: Issues in evaluation of stream learning algorithms. pp. 329–338 (2009). https://doi.org/10.1145/1557019.1557060
14. Gao, J., Fan, W., Han, J., Yu, P.S.: A general framework for mining concept-drifting data streams with skewed distributions. In: Proceedings of the 2007 siam international conference on data mining. pp. 3–14. SIAM (2007)
15. García, V., Sánchez, J.S., Rodríguez-Picón, L.A., Méndez-González, L.C., de Jesús Ochoa-Domínguez, H.: Using regression models for predicting the product quality in a tubing extrusion process. Journal of Intelligent Manufacturing **30**(6), 2535–2544 (2019)
16. Gavoyiannis, A., Voumvoulakis, E., Hatziargyriou, N.: On-line supervised learning for dynamic security classification using probabilistic neural networks. In: IEEE Power Engineering Society General Meeting, 2005. pp. 2669–2675 Vol. 3 (2005). https://doi.org/10.1109/PES.2005.1489656
17. Hammer, B., Hüllermeier, E.: Interpretable Machine Learning: On the Problem of Explaining Model Change. In: Proceedings – 31. Workshop Computational Intelligence. KIT Scientific Publishing, Karlsruhe (2021)
18. Hinder, F., Brinkrolf, J., Vaquet, V., , Hammer, B.: A Shape-Based Method for Concept Drift Detection and Signal Denoising. In: 2021 IEEE Symposium Series on Computational Intelligence (SSCI). IEEE (2021)
19. Hoeffding, W.: Probability Inequalities for sums of Bounded Random Variables, pp. 409–426. Springer New York, New York, NY (1994). https://doi.org/10.1007/978-1-4612-0865-5_26
20. Hoi, S.C., Sahoo, D., Lu, J., Zhao, P.: Online learning: A comprehensive survey. Neurocomputing **459**, 249–289 (2021). https://doi.org/https://doi.org/10.1016/j.neucom.2021.04.112
21. Hung, S.Y., Lee, C.Y., Lin, Y.L.: Data science for delamination prognosis and online batch learning in semiconductor assembly process. IEEE Transactions on Components, Packaging and Manufacturing Technology **10**(2), 314–324 (2020). https://doi.org/10.1109/TCPMT.2019.2956485

22. Ikonomovska, E., Gama, J., Džeroski, S.: Learning model trees from evolving data streams. Data Mining and Knowledge Discovery **23**(1), 128–168 (2010). https://doi.org/10.1007/s10618-010-0201-y
23. Jain, L., Seera, M., Lim, C., Balasubramaniam, P.: A review of online learning in supervised neural networks. Neural Computing and Applications **25**, 491–509 (2013). https://doi.org/10.1007/s00521-013-1534-4
24. Jakob, J., Hasenjaeger, M., Hammer, B.: On the suitability of incremental learning for regression tasks in exoskeleton control. In: 2021 IEEE Symposium Series on Computational Intelligence (SSCI). IEEE (2021)
25. Jena, M., Dehuri, S.: DecisionTree for classification and regression: A state-of-the art review. Informatica **44**(4) (2020). https://doi.org/10.31449/inf.v44i4.3023
26. Jeong, Y.S., Byon, Y.J., Castro-Neto, M.M., Easa, S.M.: Supervised weighting-online learning algorithm for short-term traffic flow prediction. IEEE Transactions on Intelligent Transportation Systems **14**(4), 1700–1707 (2013). https://doi.org/10.1109/TITS.2013.2267735
27. Lakshminarayanan, B., Roy, D., Teh, Y.: Mondrian forests for large-scale regression when uncertainty matters. In: 19th International Conference on Artificial Intelligence and Statistics (AISTATS) 2016. vol. 51 (2016)
28. Loh, W.Y.: Regression trees with unbiased variable selection and interaction detection. Statistica Sinica **12**, 361–386 (2002)
29. Losing, V., Hammer, B., Wersing, H.: Incremental on-line learning: A review and comparison of state of the art algorithms. Neurocomputing **275**, 1261–1274 (2018). https://doi.org/10.1016/j.neucom.2017.06.084
30. Manapragada, C., Webb, G., Salehi, M.: Extremely fast decision tree. pp. 1953–1962 (2018). https://doi.org/10.1145/3219819.3220005
31. Mouss, H., Mouss, D., Mouss, N., Sefouhi, L.: Test of page-hinckley, an approach for fault detection in an agro-alimentary production system. In: 2004 5th Asian Control Conference (IEEE Cat. No.04EX904). vol. 2, pp. 815–818 Vol.2 (2004)
32. Murata, N., Kawanabe, M., Ziehe, A., Müller, K.R., ichi Amari, S.: On-line learning in changing environments with applications in supervised and unsupervised learning. Neural Networks **15**(4), 743–760 (2002). https://doi.org/https://doi.org/10.1016/S0893-6080(02)00060-6
33. Murthy, S., Salzberg, S.: Lookahead and pathology in decision tree induction. In: IJCAI. pp. 1025–1033. Citeseer (1995)
34. Nelles, O.: Nonlinear System Identification. Springer International Publishing (2020). https://doi.org/10.1007/978-3-030-47439-3
35. Saffari, A., Leistner, C., Santner, J., Godec, M., Bischof, H.: On-line random forests. In: 2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops. pp. 1393–1400 (2009). https://doi.org/10.1109/ICCVW.2009.5457447
36. Tax, D., Laskov, P.: Online svm learning: from classification to data description and back. In: 2003 IEEE XIII Workshop on Neural Networks for Signal Processing (IEEE Cat. No.03TH8718). pp. 499–508 (2003). https://doi.org/10.1109/NNSP.2003.1318049
37. Zhang, N., Wu, Q.: Online learning for supervised dimension reduction. Mathematical Foundations of Computing **2**(2), 95–106 (2019)

Spectrum-Revealing CUR Decomposition for Sparse Matrices

Onyebuchi Ekenta and Ming Gu

University of California Berkeley , Berkeley CA 94720, USA

Abstract. The CUR decomposition is a popular tool for computing a low rank factorization of a matrix in terms of a small number of columns and rows of the matrix. CUR decompositions are favored in some use-cases because they have a higher degree of interpretability and are able to preserve the sparsity of the input matrix. Previous random sampling-based approaches are able to construct CUR decompositions with relative-error bounds with high probability. However, these methods are costly to run on practical datasets. We implement a novel algorithm to compute CUR approximations of sparse matrices. Our method comes with relative error bounds for matrices with rapidly decaying spectrum and runs in time that is nearly linear in m and n.

Keywords: CUR Decomposition · Low Rank Approximation · Sparse Matrix

1 Introduction

Data analysis is essential to making scientific progress in the modern world. Scientists often rely on data collected through massive experiments involving hundreds of sensors or produced through large simulations to gain insights the structure and behavior of the systems they study. These datasets can reach enormous sizes. Experiments from the LHC can produce petabytes worth of data. The ITER project, the world's largest nuclear fusion project, is expected to produce two petabytes of data every day by the year 2035. Analyzing such data sets often necessitates the use of considerable computing resources. Access to and useage of such computing platforms is a limiting factor for scientific programs conducted across a wide range of disciplines. Having access to more scalable and efficient procedures for data analysis provides researchers with greater flexibility by allowing them to analyze their experiments faster and with more easily accessible computational resources.

Low rank approximation is a common technique in data analysis for reducing the noise and understanding the relationship between data variables. A common approach to producing low rank approximations is computing a truncated singular value decomposition. While this method provides the most accurate approximations, it has certain drawbacks that make it unsuitable for certain applications. The resulting singular value vectors will typically be dense even for

sparse input matrices which can result in excessive costs in storage and processing time. Also, it is sometimes desirable to interpret the singular value vectors as though they were instances of the data set (e.g. eigengenes for gene-based data). But this interpretation becomes difficult when using the SVD as the singular value vectors do not conserve important properties of the matrix such as nonnegativity and sparsity.

The CUR matrix decomposition is an alternative approach that is better suited for handling these issues. CUR decomposition computes a low rank approximation of the form $\mathbf{A} \approx \mathbf{CUR}$ where the matrices $\mathbf{C}$ and $\mathbf{R}$ consist of columns and rows $\mathbf{A}$. By approximating the matrix in terms of actual columns and rows it both preserves the sparsity of the original matrix and allows for the natural interpretation as instances of the data set. Because of these properties, CUR decompositions have become popular in the data science community where they have been applied to problems such as feature selection, clustering and graph mining. [26, 21, 5]

Some random-sampling based methods come with relative-error guarantees [10, 30, 6]. That is to say, with high probability the CUR decomposition will satisfy

$$\|\mathbf{A} - \mathbf{CUR}\|_F^2 < (1+\epsilon)\|\mathbf{A} - \mathbf{A}_k\|_F^2$$

where $\mathbf{A}_k$ is the optimal rank k approximation. However these methods require computing sampling $O(k/\epsilon)$ columns and rows to achieve their error guarantees which is often impractical. Some of these methods rely on quantities that are expensive to compute such as singular value vectors and leverage scores. Computing these quantities for large-scale datasets of interest can require the use of large parallel or distributed platforms. [13]

In this paper we introduce the Spectrum-Revealing CUR decomposition method (SR-CUR). This algorithm allows for the computation of relative-error CUR decompositions of matrices with a rapidly decaying spectrum. Importantly, for sparse matrices, the algorithm runs in time that is nearly linear in m and n, making it feasible to compute large factorizations with modest computing resources. We run experiments verifying the speed and accuracy of our factorizations. Our code is available on Github [11].

2 Related Work

2.1 CUR decomposition

A CUR decomposition approximates a matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ as the product of a matrix $\mathbf{C} \in \mathbb{R}^{m \times c}$ consisting of a collection of columns of $\mathbf{A}$, $\mathbf{R} \in \mathbb{R}^{r \times n}$ consisting of a collection of rows of $\mathbf{A}$ and an inner mixing matrix $\mathbf{U}$. The goal is to find a choice of columns and rows which minimizes the factorization error $\|\mathbf{A} - \mathbf{CUR}\|$.

A variety of different approaches have been taken to selecting the columns and rows for the factorization. One approach is to employ deterministic pivoting strategies to make the selections [25, 7, 17, 3]. Another useful approach is to

employ the maximum volume principle, meaning one seeks to select the columns and rows that maximizes the absolute determinant of the matrix formed at the intersection of C and R. This principle was employed to develop a CUR decomposition method known as pseudoskeleton approximation [16, 15, 14]. In [27] a similar principle known as simplex volume maximization is used.

In [9] the authors propose a linear time CUR decomposition algorithm. A collection of c columns and r rows are randomly sampled to construct the C and R matrices which are in turn used to compute U. For a given target rank k, by sampling $c = O(\log(1/\delta)\epsilon^{-4})$ columns and $r = O(k\delta^{-2}\epsilon^{-2})$ rows the CUR decomposition will satisfy

$$\|A - CUR\|_2 \leq \|A - A_k\|_2 + \epsilon\|A\|_2$$

with probability $1 - \delta$. Alternatively by sampling $c = O(k\log(1/\delta)\epsilon^{-4})$ columns and $r = O(k\delta^{-2}\epsilon^{-2})$ rows the above will hold for the Frobenius norm with probability $1 - \delta$.

This method was improved in [10] to yield a relative error factorization method. Here C and R are randomly sampled and U is set to be the Moore-Penrose pseudoinverse of their intersection. The sampling probabilities used here depend on the right-singular vectors of A. In [21] a random-sampling based CUR decomposition method was presented which employed statistical leverage scores to construct the sample distribution. In [6] and also in [30] relative-error CUR decompositions running in linear time and requiring $O(k/\epsilon)$ row and column samples are presented.

The RandomizedSVD algorithm introduced in [22, 1] can approximate the singular vectors in $O(mn\log k)$ time. In [28, 29] factorization methods were proposed achieving expected relative error and improved performance compared to existing methods. See [19] for more information.

2.2 Rank Revealing Factorizations

Rank-revealing algorithms [7] are low rank matrix approximation algorithms that accurately capture the rank of a given matrix. One such example is the rank revealing QR factorization. Given an $m \times n$ matrix A this produces a factorization of the form

$$AP = QR = Q\begin{bmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{bmatrix}$$

where $Q \in \mathbb{R}^{m\times n}$ is orthonormal, $R_{11} \in \mathbb{R}^{k\times k}$, $R_{12} \in \mathbb{R}^{k\times n-k}$ and $R_{22} \in \mathbb{R}^{n-k\times n-k}$. The factorization is called rank-revealing if

$$\frac{\sigma_k(A)}{p(k,n)} \leq \sigma_{min}(R_{11}) \leq \sigma_k(A)$$
$$\sigma_{k+1}(A) \leq \sigma_{\max}(R_{22}) \leq p(k,n)\sigma_{k+1}(A)$$

where $p(k,n)$ is a low-degree polynomial in k and n. A low rank factorization can be obtained from RRQR by neglecting R_{22}. The resulting low rank factorization $\tilde{A}$ satisfies

$$\|A - \tilde{A}\|_2 \leq p(k,n)\sigma_{k+1}(A)$$

. A table of achievable values for $p(k,n)$ can be found in [4]. Rank-revealing methods need not capture the full spectrum of the of the matrix. Mirian and Gu introduced a new low rank approximation scheme that approximated the full spectrum of the matrix. [23]

2.3 Randomized Sketching

Sketching is a technique where a large problem is replaced by a much smaller which can inform the solution of the original problem. A good example of how randomization can help improve algorithms in numerical linear algebra can be observed in the problem of linear regression. Consider an $m \times n$ matrix $\mathbf{A}$ with $n \ll m$. Since the system is overdetermined the problem $A\mathbf{x} = \mathbf{b}$ cannot be solved exactly so we seek to solve the minimization problem $\min_{\mathbf{x}} \|\mathbf{Ax} - \mathbf{b}\|$. This problem can be greatly reduced in size by applying a matrix $\mathbf{\Omega}$ that is randomly sampled from some distribution to produce the much smaller problem problem $\min_{\mathbf{x}} \|\mathbf{\Omega Ax} - \mathbf{\Omega b}\|$. In some cases, solving this smaller system yields an approximate solution to the original problem.

The sampling matrix $\mathbf{\Omega}$ can be constructed in a variety of different, the simplest being forming it out of independent Gaussian distributions. One can also construct sparse linear maps or maps that have specialized internal structure. [2]

In addition to least-squares problems, randomized techniques have also gained popularity for the problem of low-rank approximation, where certain methods improve upon their deterministic counterparts. [31, 20, 18].

3 Our Approach

3.1 Overview

Given an $m \times n$ matrix $\mathbf{A}$ our algorithm begins by computing a truncated LU factorization of the matrix. More precisely, for a choice of rank ℓ, we compute a factorization of the form

$$\mathbf{PAQ} = \begin{pmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{pmatrix} = \begin{pmatrix} \mathbf{L}_{11} & \\ \mathbf{L}_{21} & \mathbf{I}_{n-k} \end{pmatrix} \begin{pmatrix} \mathbf{U}_{11} & \mathbf{U}_{12} \\ & \mathbf{S} \end{pmatrix}$$

where the submatrix $\mathbf{A}_{11}$ is $\ell \times \ell$. The factorizations are computed using the LUSOL software package [12]. LUSOL provides implementations of threshold complete pivoting (TCP) and threshold rook pivoting (TRP) pivot strategies

which use a Markowitz strategy to maintain sparsity of the matrix. Either can be used to produce the initial factorization. This LU decomposition provides us with our initial selection of columns and rows, namely $\mathbf{C} = \begin{pmatrix} \mathbf{A}_{11} \\ \mathbf{A}_{21} \end{pmatrix}$ and $\mathbf{R} = \begin{pmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \end{pmatrix}$. We also retain the Schur complement $\mathbf{S}$ which will be used in the Spectrum-Revealing Pivoting procedure described later.

After the initial factorization the $\mathbf{L}_{21}$ and $\mathbf{U}_{12}$ matrices are discarded and we are left storing only the factorization $\mathbf{A}_{11} = \mathbf{L}_{11}\mathbf{U}_{11}$. Following this, we run the Spectrum-Revealing Pivoting (SRP) procedure to improve the quality of the factorization. The final CUR decomposition will be computed with the StableCUR procedure.

3.2 Estimating Element with the Largest Magnitude

Computing the maximum entry of a matrix can be expensive if we only have indirect access to the matrix, such as through a factorization $\mathbf{A} = \mathbf{BC}$. We implement a method to estimate the maximum value in such cases at a reduced cost. We begin by producing a random matrix sketch via $\mathbf{R} = \mathbf{\Omega A}$ where $\mathbf{\Omega}$ is a $p \times m$ random matrix whose entries are sampled independently from the standard normal distribution. EELM then determines the column of $\mathbf{R}$ of maximum norm and returns the maximum element in the corresponding column of $\mathbf{A}$. When p is small computing the p matrix vector products to compute $\mathbf{\Omega A}$ can be more efficient than computing all entries of $\mathbf{A}$.

Using EELM we are able to quickly identify large elements of the Schur complement which can be used to improve the quality of the factorization. In addition to this, if the value returned by EELM is small we can bound the error of the factorization with high probability. The quality of the estimated maximum is given by the following theorem

Theorem 1. *For $p = \Theta\left(\log(n/\delta)/\epsilon^{-2}\right)$ the estimated value x given by Algorithm 1 satisfies the inequality $x \geq \sqrt{\frac{1-\varepsilon}{mn(1+\varepsilon)}}\|\mathbf{A}\|_F$ with probability at least $1 - \delta$.*

In particular, we only require $p = O(\log n)$ to yield good approximations with high probability. In practice p is set to some reasonable constant (e.g. 20).

3.3 Spectrum Revealing Pivoting

In the SRP procedure we identify rows and columns that can be swapped with the rows and columns chosen during the truncated LU facotorization procedure to improve the quality of the factorization. We apply the maximum volume principle as a heuristic to judge the quality of a selection of rows and columns. Thus, we perform a sequence of row and column swaps with the goal of maximizing $|\det \mathbf{A}_{11}|$. Each swap will transform $\mathbf{A}_{11}$ into a new $\ell \times \ell$ submatrix $\mathbf{A}'_{11}$ whose volume is at least a factor $f > 1$ greater than the volume of $\mathbf{A}_{11}$, where f is a

Algorithm 1: Estimating Element with Largest Magnitude (EELM)

Input: A matrix $A \in \mathbb{R}^{m \times n}$ and its projection $\mathbf{R} \in \mathbb{R}^{p \times n}$
Output: r, c, m - row index, column index, and value of the largest element

1 $c = \underset{i<j<n}{\arg\max} \|\mathbf{R}(:, j)\|_2$
2 Compute the c-th column of $\mathbf{A}$
3 $r = \underset{i \leq j \leq m}{\arg\max} |\mathbf{A}(j, c)|$
4 $m = \mathbf{A}(r, c)$

tolerance parameter. We repeat this process until no appropriate swap can be found.

We determine the swap in two phases. First we extend $\mathbf{A}_{11}$ into a $(\ell+1) \times (\ell+1)$ matrix $\overline{\mathbf{A}}_{11}$ by adding a new row $\ell+i$ and column $\ell+j$ to the matrix $\mathbf{A}_{11}$, forming a $(\ell+1) \times (\ell+1)$ matrix $\overline{\mathbf{A}}_{11}$. Then we choose a row $i' \in \{1, \ldots, \ell, \ell+i\}$ and column $j' \in \{1, \ldots, \ell, \ell+i\}$ we wish to remove from $\overline{\mathbf{A}}_{11}$.

Having identified the relevant rows and columns we swap row $\ell+i$ with i' and column $\ell+j$ with column j', thus transforming $\mathbf{A}_{11}$ into a new $\ell \times \ell$ matrix $\mathbf{A}'_{11}$. Note that if either $i' = \ell+i$ or $j' = \ell+j$ then the corresponding swap need not be performed. The following theorem helps guide the selection of the rows and columns.

Theorem 2. *Let* $\alpha = \mathbf{S}(i, j)$ *and* $\beta = \overline{\mathbf{A}}_{11}^{-T}(i', j')$. *Then we have,*

$$\left| \frac{\det \mathbf{A}'_{11}}{\det \mathbf{A}_{11}} \right| = |\alpha\beta|$$

.

Thus we have that if $|\alpha\beta| > f$ we know the volume of $\mathbf{A}_{11}$ will increase by at least a factor of f. Ideally, to select the appropriate rows of and columns we'd want to identify the maximum elements of $\mathbf{S}$ and $\overline{\mathbf{A}}_{11}^{-T}$ so as to maximize α and β. However, computing the exact maximum would be too computationally expensive, so we apply EELM to estimate them. To do this SRP must keep track of matrix sketches of $\mathbf{A}_{11}^{-T}$ and $\mathbf{S}$.

After identifying the swap to be performed the LU factorization and the matrix sketches must be updated to reflect the swap. This means we must update the LU factorization of $\mathbf{A}_{11}$ and the matrix sketches of $\mathbf{S}$ and $\overline{\mathbf{A}}_{11}^{-T}$. The row and column swaps can be implemented with LUSOL's column replacement and row replacement routines. Since $\overline{\mathbf{A}}_{11}^{-T}$ is small it is cheap to simply recompute a new random projection each step. For the Schur complement we iteratively update the projection at each step. Let $\mathbf{\Omega}$ be the random $p \times m - \ell$ matrix used to compute the projection of $\mathbf{S}$. In each iteration we compute the projection of the new Schur complement $\mathbf{\Omega S}'$ as an update to the previous projection $\mathbf{\Omega S}$. Details are provided in the next section.

Updating Sketched Schur Complement

Algorithm 2: Spectrum Revealing Pivoting

Input: f - tolerance parameter
$\tilde{\mathbf{A}}_{11}^{-T}$ - $q \times (\ell + 1)$ Sketch of $\overline{\mathbf{A}}_{11}^{-T}$
$\tilde{\mathbf{S}}$ - $p \times n$ Sketch of Schur Complement

while *True* **do**
  $(i, j, \alpha) \leftarrow EELM(\mathbf{S}, \tilde{\mathbf{S}})$
  Form $\overline{\mathbf{A}}_{11} = \mathbf{A}([1:k, k+i], [1:k, k+j])$
  $(i', j', \beta) \leftarrow EELM(\mathbf{A}_{11}^{-T}, \tilde{\mathbf{A}}_{11}^{-T})$
  if $|\alpha\beta| < f$ **then**
    break
  end
  Swap rows $k+i$ and i' and columns $k+j$ and j' and update the LU factorization.
  Update $\tilde{\mathbf{S}}$
  Recompute random sketch $\tilde{\mathbf{A}}_{11}^{-T}$
end

Rank One Update In the cases where only a single swap is performed the swap corresponds to a rank one update of the matrix. That is we have that

$$\mathbf{A}' = \mathbf{A} + \mathbf{v}\mathbf{w}^T$$

for the appropriate choice of $\mathbf{v}$ and $\mathbf{w}$. For example if we are swapping columns j_1 and j_2 then $\mathbf{v} = \mathbf{A}(:, j_2) - \mathbf{A}(:, j_1)$ and $w = \mathbf{e}_{j_1} - \mathbf{e}_{j_2}$ where $\mathbf{e}_k$ is the k-th standard basis vector of $\mathbb{R}^n$. The following theorem allows us describe the update to the Schur complement when the matrix goes under a rank one update.

Theorem 3. *Let* $\mathbf{B} = \mathbf{A} + \mathbf{v}\mathbf{w}^T$. *We partition* $\mathbf{B}$, $\mathbf{v}$ *and* $\mathbf{w}$ *according to the partion for* $\mathbf{A}$. *Let* $\mathbf{S}' = \mathbf{B}_{22} - \mathbf{B}_{21}\mathbf{B}_{11}^{-1}\mathbf{B}_{12}$. *Then we have* $\mathbf{S}' = \mathbf{S} + \mathbf{E}$ *where*

$$\begin{aligned}
\mathbf{E} &= \frac{1}{d}\hat{\mathbf{v}}\hat{\mathbf{w}}^T \\
\hat{\mathbf{v}} &= \left(\mathbf{v}_2 - \mathbf{A}_{21}\mathbf{A}_{11}^{-1}\mathbf{v}_1\right) \\
\hat{\mathbf{w}}^T &= \left(\mathbf{w}_2^T - \mathbf{w}_1^T\mathbf{A}_{11}^{-1}\mathbf{A}_{12}\right) \\
d &= \left(1 + \mathbf{w}_1^T\mathbf{A}_{11}^{-1}\mathbf{v}_1\right)
\end{aligned}$$

With this, the corresponding update to the sketched Schur complement can be computed via

$$\mathbf{\Omega}\mathbf{S}' = \mathbf{\Omega}\mathbf{S} + \frac{1}{d}\mathbf{\Omega}\hat{\mathbf{v}}\hat{\mathbf{w}}^T$$

It takes $O(\ell^2)$ steps to compute d. Computing $\hat{\mathbf{v}}$ can be done in $O(m\ell)$ time and computing $\hat{\mathbf{w}}$ takes $O(n\ell)$ time. Computing the quantity $\frac{1}{d}\mathbf{\Omega}\hat{\mathbf{v}}\mathbf{w}^T$ and adding

it to $\mathbf{\Omega S}$ requires an additional $O(mp + np)$ steps. So the update to the Schur complement takes $O((\ell + p)(m + n))$ time to compute.

Rank Two Update Now we consider the case where rows $\ell+i$ and i' and columns $\ell + j$ and j' are swapped. First we must define some vectors. Let $\mathbf{a}_{21}^T$ and $\mathbf{a}_{12}$ be i-th row of $\mathbf{A}_{21}$ and the j-th column of $\mathbf{A}_{12}$ respectively. Let $\mathbf{s}_c$ and $\mathbf{s}_r^T$ be the the i-th column and the j-th row of the Schur complement. Let $\mathbf{e}_{i'}$ be the i'-th standard basis vector in $\mathbb{R}^\ell$ and $\overline{\mathbf{e}}_i$ be the i-th standard basis vector of $\mathbb{R}^{m-\ell}$. Let $\mathbf{e}_{j'}^T$ be the j'-th standard basis (row) vector of $\mathbb{R}^\ell$ and and $\overline{\mathbf{e}}_j^T$ be he j-th standard (row) vector of $\mathbb{R}^{n-\ell}$. Finally, let $\alpha = \mathbf{S}(i, j)$ and $\beta = \overline{\mathbf{A}}_{11}^{-T}(i', j')$, where $\overline{\mathbf{A}}_{11}$ is as defined previously. Then we have the following theorem characterizing the update to $\mathbf{S}$.

Theorem 4. *After swapping rows $\ell + i$ and i' and columns $\ell + j$ and j' the new Schur complement will be given by*

$$\mathbf{S}' = \mathbf{S} - \frac{1}{\alpha}\mathbf{s}_c\mathbf{s}_r^T + \frac{1}{\beta}\mathbf{v}_c\mathbf{v}_r^T$$

, where $\mathbf{v}_c$ and $\mathbf{v}_r$ are given by

$$\mathbf{v}_c = \mathbf{A}_{21}\mathbf{A}_{11}^{-1}\mathbf{e}_{i'} + \overline{\mathbf{e}}_i + \frac{\mathbf{a}_{21}^T\mathbf{A}_{11}^{-1}\mathbf{e}_{i'}}{\alpha}\mathbf{s}_c$$
$$\mathbf{v}_r^T = \mathbf{e}_{j'}^T\mathbf{A}_{11}^{-1}\mathbf{A}_{12} + \overline{\mathbf{e}}_j^T + \frac{\mathbf{e}_{j'}^T\mathbf{A}_{11}^{-1}\mathbf{a}_{12}}{\alpha}\mathbf{s}_r^T$$

Given this result we can update the sketch of the Schur complement via

$$\Omega\mathbf{S}' = \Omega\mathbf{S} - \frac{1}{\alpha}\Omega\mathbf{s}_c\mathbf{s}_r^T + \frac{1}{\beta}\Omega\mathbf{v}_c\mathbf{v}_r^T$$

.

Computing the vectors $\mathbf{s}_c$ and $\mathbf{v}_c$ can be done in $O(m\ell)$ time and computing $\mathbf{s}_r$ and $\mathbf{v}_r$ can be done in $O(n\ell)$ time. The corresponding update to the sketched Schur complement will require an additional $O(mp+np)$ steps to compute. The total time necessary to update the Schur complement is therefore $O((\ell+p)(m+n))$.

3.4 Spectrum-Revealing CUR

The end result of SRLU gives a selection of columns and rows to use for the CUR decomposition. The final CUR decomposition is given by the StableCUR procedure described in Algorithm 3 with the above choice of $\mathbf{C}$ and $\mathbf{R}$ as input. Since this method requires the computation of QR factorizations, it comes with an increased memory cost.

Algorithm 3: StableCUR

Input: A matrix $A \in \mathbb{R}^{m \times n}$, $\mathbf{R} \in \mathbb{R}^{\ell \times n}$ $\mathbf{C} \in \mathbb{R}^{m \times \ell} \in$
Output: $\tilde{\mathbf{A}}_k$

1 Do QR factorization on $\mathbf{R}^T$ to obtain a basis of rows of $\mathbf{R}, \mathbf{R} = \mathbf{R}_r\mathbf{Q}_r$
2 Do QR factorization on $\mathbf{C}$ to obtain a basis of columns of $\mathbf{C}, \mathbf{C} = \mathbf{Q}_c\mathbf{R}_c$
3 $\mathbf{B} = \mathbf{Q}_c^T\mathbf{A}\mathbf{Q}_r^T$
4 Do SVD on $\mathbf{B}$ to Compute $\mathbf{B}_k$
5 $\widetilde{\mathbf{A}}_\mathbf{k} = \mathbf{Q}_c\mathbf{B}_k\mathbf{Q}_r$

4 Theoretical Analysis

4.1 Time Complexity

LU Decomposition For dense matrices the truncated LU decomposition would take $O(\ell mn)$ time. For sparse matrices the time varies depending on the size and degree of sparsity of the matrix.

The choice of TCP and TRP affects the overall run time of the algorithm. When threshold complete pivoting is used, typically the follow-up spectrum-revealing pivoting steps typically performing very few or no swaps at all. As a result TCP will typically be the faster choice for smaller matrices. However if the matrix is too large or too dense TCP can become too expensive to compute and so TRP will be the better choice.

When TRP is used, the spectrum-revealing pivoting tends to perform many swaps and he initial factorization will typically only represent a small fraction of the total computation time. Thus the total computation time will typically be dominated by the SRP and StableCUR procedures.

Spectrum-Revealing Pivoting Running EELM for the Schur complement requires $O(pn)$ time. Recomputing the sketch $\tilde{\mathbf{A}}_{11}$ and estimating the maximum of $\mathbf{A}_{11}^{-T}$ requires $O(q\ell^2)$ time. The LUSOL routines to update the factorization require $O(\ell^2)$ steps. The update of the projected Schur complement requires $O((p+\ell)(m+n))$ steps. Thus the total time for an iteration of one iteration of SRP takes $O(q\ell^2 + (p+\ell)(m+n))$ time.

With a probability of .99 SRP will require at most $O(\ell \log(mn))$ iterations [8]. Thus the total complexity for spectrum-revealing pivoting is $O(\ell \log(mn)(q\ell^2 + (p+\ell)(m+n))$. EELM procedure yields good approximations for $p = O(\log n)$ and $q = O(\log \ell)$. Thus the total time complexity of the Spectrum-Revealing Pivoting method is $O\left(\ell \log(mn)(\ell^2 \log \ell + (\log n + \ell)(m+n))\right)$.

StableCUR The QR factorization of $\mathbf{R}^T$ and $\mathbf{C}$ requires $O(m\ell^2)$ time and $O(n\ell^2)$ time respectively to compute. For dense matrices $\mathbf{B} = \mathbf{Q}_c^T\mathbf{A}\mathbf{Q}_r^T$ would require $O(mn\ell)$ time. But since $\mathbf{A}$ is sparse significant savings are made. If A is sparse, the matrix-vector product $A\mathbf{v}$ can be computed in time $O(\text{nnz}(A))$.

Computing $\tilde{\mathbf{A}} = \mathbf{A}\mathbf{Q}_r^T$ can be done in $O(\text{nnz}(A)\ell)$ time. Then, computing $\mathbf{Q}_c^T\tilde{\mathbf{A}}$ requires an additional $O(m\ell^2)$ steps to compute.

4.2 Error Bounds

Spectrum Revealing Pivoting

Theorem 5. *For $j \leq k$ and $\gamma = O(fk\sqrt{mn})$, SRP produces a rank k SRLU factorization with*

$$\left\| \Pi_1 \mathbf{A} \Pi_2^T - \hat{\mathbf{L}}\hat{\mathbf{U}} \right\|_2 \leq \gamma \sigma_{k+1}(\mathbf{A})$$
$$\left\| \Pi_1 \mathbf{A} \Pi_2^T - (\widehat{\mathbf{L}}\widehat{\mathbf{U}})_j \right\|_2 \leq \sigma_{j+1}(\mathbf{A}) \left(1 + 2\gamma \frac{\sigma_{k+1}(\mathbf{A})}{\sigma_{j+1}(\mathbf{A})}\right)$$

SR-CUR

Theorem 6. *For $\gamma = O(fk\sqrt{mn})$ the SR-CUR decomposition satisfies*

$$\|\mathbf{A} - \widehat{\mathbf{A}}\|_2 \leq \|\mathbf{A} - \widehat{\mathbf{A}}\|_F \leq \gamma \sigma_{\ell+1}(\mathbf{A}),$$
$$\left\|\mathbf{A} - \widehat{\mathbf{A}}_k\right\|_F^2 \leq \left(1 + \frac{2\gamma^2 \sigma_{\ell+1}^2(\mathbf{A})}{\sum_{i=k+1}^{\text{rank}(\mathbf{A})} \sigma_i^2(\mathbf{A})}\right) \|\mathbf{A} - \mathbf{A}_k\|_F^2,$$
$$\left\|\mathbf{A} - \widehat{\mathbf{A}}_k\right\|_2^2 \leq \left(1 + 2\gamma^2 \frac{\sigma_{\ell+1}^2(\mathbf{A})}{\sigma_{k+1}^2(\mathbf{A})}\right) \|\mathbf{A} - \mathbf{A}_k\|_2^2,$$

with probability .98

5 Experiments

5.1 SR-CUR

We provide an experiment demonstrating the capacity of SR-CUR to compute nearly optimal CUR decompositions. Drawing from the SuiteSparse Matrix Collection we collect a dataset consisting of matrices exhibiting rapidly decaying spectrum.

We compute factorizations for $k = 100, 200, 300, 400, 500$. We use TRP for the initial factorization and set $f = 2$. For each k we set $\ell = k + 50$ to apply the Stable-CUR algorithm. The SVD computations were computed with Matlab's `svds` function for $k = 500$. We compute the relative Frobenius error $\|A - A_k\|_F / \|A\|_F$ for each factorization. We compare the results against SVD and the L2-norm based random sampling method described in [9] with implementation provided by [24]. The results are shown in Figure 1. The time spent on the $k = 500$ factorization is shown in Figure 2. To simulate a low resource environment the maximum number of MATLAB threads was set to 4 during these computations.

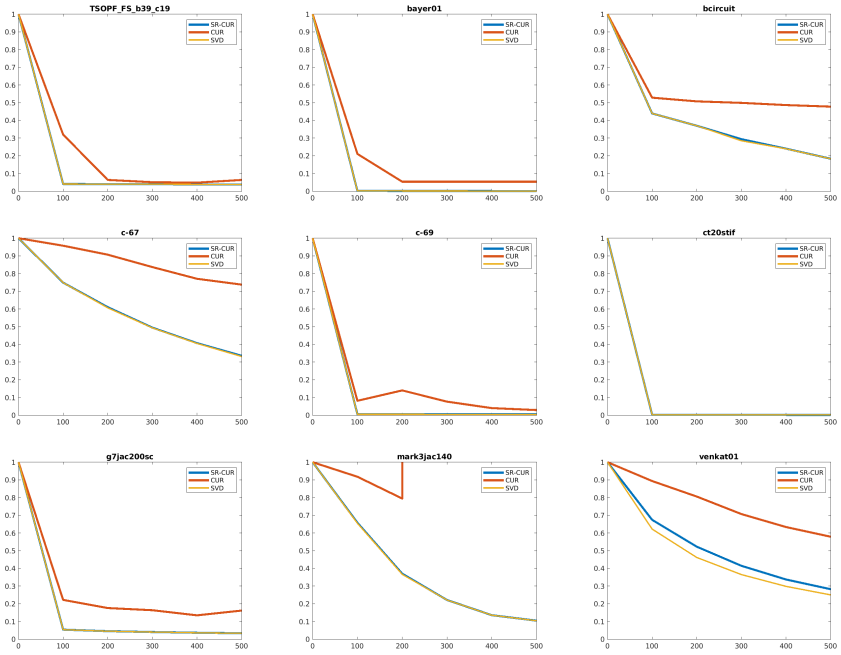


Fig. 1. Comparison of SVD, SR-CUR and L2-norm sampling CUR

6 Conclusion

We provide a high quality implementation of a the SR-CUR decomposition algorithm. We see that for matrices with rapidly decaying spectrum SR-CUR is capable of achieving releative-error bounds. For sparse matrices, our approach comes with improved complexity bounds that allow it to scale easily to to large matrices without the use of parallelization. Our experiments verify that our approach offers substantial benefits over alternative methods.

References

1. A randomized algorithm for the decomposition of matrices. Applied and Computational Harmonic Analysis **30**(1), 47–68 (2011). https://doi.org/https://doi.org/10.1016/j.acha.2010.02.003, https://www.sciencedirect.com/science/article/pii/S1063520310000242
2. Ailon, N., Chazelle, B.: The fast johnson–lindenstrauss transform and approximate nearest neighbors. SIAM Journal on Computing **39**(1), 302–322 (2009). https://doi.org/10.1137/060673096, https://doi.org/10.1137/060673096
3. Berry, M.W., Pulatova, S.A., Stewart, G.W.: Algorithm 844: Computing sparse reduced-rank approximations to sparse matrices. ACM Trans. Math. Softw. **31**(2), 252–269 (Jun 2005). https://doi.org/10.1145/1067967.1067972, https://doi.org/10.1145/1067967.1067972

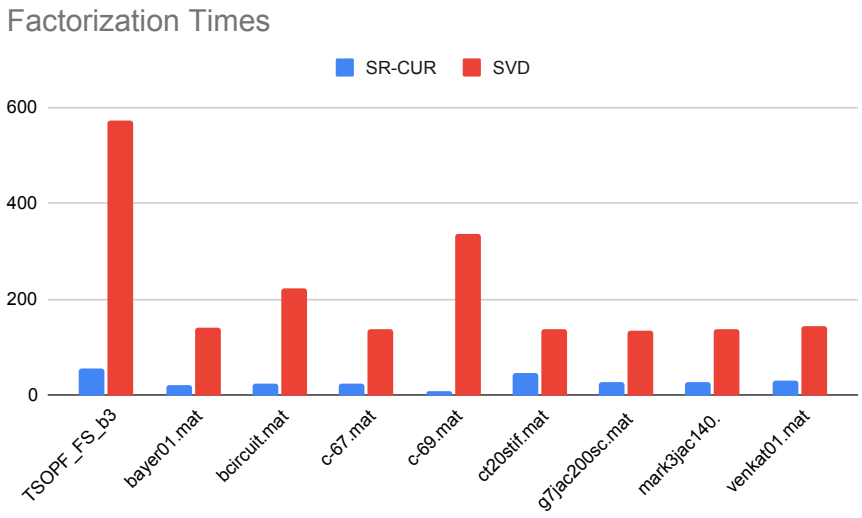


Fig. 2. Computation time in seconds for Rank 500 Factorizations

4. Boutsidis, C., Mahoney, M., Drineas, P.: On selecting exactly k columns from a matrix (01 2008)
5. Boutsidis, C., Mahoney, M.W., Drineas, P.: Unsupervised feature selection for principal components analysis. In: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. p. 61–69. KDD '08, Association for Computing Machinery, New York, NY, USA (2008). https://doi.org/10.1145/1401890.1401903, https://doi.org/10.1145/1401890.1401903
6. Boutsidis, C., Woodruff, D.P.: Optimal cur matrix decompositions. SIAM Journal on Computing **46**(2), 543–589 (2017). https://doi.org/10.1137/140977898, https://doi.org/10.1137/140977898
7. Chan, T.F.: Rank revealing qr factorizations. Linear Algebra and its Applications **88-89**, 67–82 (1987). https://doi.org/https://doi.org/10.1016/0024-3795(87)90103-0, https://www.sciencedirect.com/science/article/pii/0024379587901030
8. Chen, C., Gu, M., Zhang, Z., Zhang, W., Yu, Y.: Efficient spectrum-revealing cur matrix decomposition. In: Chiappa, S., Calandra, R. (eds.) Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics. Proceedings of Machine Learning Research, vol. 108, pp. 766–775. PMLR (26–28 Aug 2020), https://proceedings.mlr.press/v108/chen20a.html
9. Drineas, P., Kannan, R., Mahoney, M.W.: Fast monte carlo algorithms for matrices iii: Computing a compressed approximate matrix decomposition. SIAM Journal on Computing **36**(1), 184–206 (2006). https://doi.org/10.1137/S0097539704442702, https://doi.org/10.1137/S0097539704442702
10. Drineas, P., Mahoney, M.W., Muthukrishnan, S.: Relative-error cur matrix decompositions. SIAM Journal on Matrix Analysis and Applications **30**(2), 844–881 (2008). https://doi.org/10.1137/07070471X, https://doi.org/10.1137/07070471X

11. Ekenta, O.: sr-cur. https://github.com/Rioghasarig/sr-cur (2022)
12. Gill, P.E., Murray, W., Saunders, M.A., Wright, M.H.: Maintaining lu factors of a general sparse matrix. Linear Algebra and its Applications **88-89**, 239–270 (1987). https://doi.org/https://doi.org/10.1016/0024-3795(87)90112-1, https://www.sciencedirect.com/science/article/pii/0024379587901121
13. Gittens, A., Devarakonda, A., Racah, E., Ringenburg, M.F., Gerhardt, L., Kottalam, J., Liu, J., Maschhoff, K.J., Canon, S., Chhugani, J., Sharma, P., Yang, J., Demmel, J., Harrell, J., Krishnamurthy, V., Mahoney, M.W., Prabhat: Matrix factorization at scale: a comparison of scientific data analytics in spark and C+MPI using three case studies. CoRR **abs/1607.01335** (2016), http://arxiv.org/abs/1607.01335
14. Goreinov, S., Oseledets, I., Savostyanov, D., Tyrtyshnikov, E., Zamarashkin, N.: How to find a good submatrix (2010)
15. Goreinov, S., Tyrtyshnikov, E., Zamarashkin, N.: A theory of pseudoskeleton approximations. Linear Algebra and its Applications **261**(1), 1–21 (1997). https://doi.org/https://doi.org/10.1016/S0024-3795(96)00301-1, https://www.sciencedirect.com/science/article/pii/S0024379596003011
16. Goreinov, S., Tyrtyshnikov, E.: The maximal-volume concept in approximation by low-rank matrices. Contemporary Mathematics **208** (01 2001). https://doi.org/10.1090/conm/280/4620
17. Gu, M., Eisenstat, S.C.: Efficient algorithms for computing a strong rank-revealing qr factorization. SIAM Journal on Scientific Computing **17**(4), 848—-869 (1995). https://doi.org/https://doi.org/10.1137/0917055
18. Halko, N., Martinsson, P.G., Tropp, J.A.: Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. SIAM Review **53**(2) (2011)
19. Kumar, N.K., Shneider, J.: Literature survey on low rank approximation of matrices. ArXiv **abs/1606.06511** (2016)
20. Mahoney, M.W.: Randomized algorithms for matrices and data. Found. Trends Mach. Learn. **3**(2), 123–224 (Feb 2011). https://doi.org/10.1561/2200000035, https://doi.org/10.1561/2200000035
21. Mahoney, M.W., Drineas, P.: Cur matrix decompositions for improved data analysis. Proceedings of the National Academy of Sciences **106**(3), 697–702 (2009). https://doi.org/10.1073/pnas.0803205106, https://www.pnas.org/content/106/3/697
22. Martinsson, P., Rohklin, V., Tygert, M.: A randomized algorithm for the approximation of matrices. Tech. rep. (2006)
23. Miranian, L., Gu, M.: Strong rank revealing lu factorizations. Linear Algebra and its Applications **367**, 1–16 (2003). https://doi.org/https://doi.org/10.1016/S0024-3795(02)00572-4, https://www.sciencedirect.com/science/article/pii/S0024379502005724
24. Schinnerl, C.: pymf3. https://github.com/charlespwd/project-title (2017)
25. Stewart, G.: Four algorithms for the the efficient computation of truncated pivoted qr approximations to a sparse matrix. Numerische Mathematikk **83**(2), 313–323 (1999). https://doi.org/https://doi.org/10.1007/s002110050451
26. Sun, J., Xie, Y., Zhang, H., Faloutsos, C.: Less is more: Compact matrix decomposition for large sparse graphs (best research paper award!). In: the 2007 SIAM International Conference on Data Mining (SDM), Minneapolis, MN (April 2007), https://www.microsoft.com/en-us/research/publication/less-is-more-compact-matrix-decomposition-for-large-sparse-graphsbest-research-paper-award/

27. Thurau, C., Kersting, K., Bauckhage, C.: Deterministic CUR for Improved Large-Scale Data Analysis: An Empirical Study, pp. 684–695. https://doi.org/10.1137/1.9781611972825.59, https://epubs.siam.org/doi/abs/10.1137/1.9781611972825.59
28. Wang, S., Zhang, Z.: A scalable cur matrix decomposition algorithm: Lower time complexity and tighter bound. In: Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1. p. 647–655. NIPS'12, Curran Associates Inc., Red Hook, NY, USA (2012)
29. Wang, S., Zhang, Z.: Improving cur matrix decomposition and the nyström approximation via adaptive sampling. J. Mach. Learn. Res. **14**(1), 2729–2769 (jan 2013)
30. Wang, S., Zhang, Z., Zhang, T.: Towards more efficient spsd matrix approximation and cur matrix decomposition. Journal of Machine Learning Research **17**(209), 1–49 (2016), http://jmlr.org/papers/v17/15-190.html
31. Woodruff, D.P.: Sketching as a tool for numerical linear algebra **10**(1–2), 1–157 (Oct 2014). https://doi.org/10.1561/0400000060, https://doi.org/10.1561/0400000060

Computing the Collection of Good Models for Rule Lists

Kota Mata[1⋆], Kentaro Kanamori[1⋆⋆], and Hiroki Arimura[1]

Graduate School of IST, Hokkaido University, Sapporo 060-0814, Japan
arim@ist.hokudai.ac.jp

Abstract. Since the seminal paper by Breiman in 2001, who pointed out a potential harm of prediction multiplicities from the view of explainable AI, global analysis of a collection of *all good models*, also known as a "*Rashomon set*," has been attracted much attention for the last years. Since finding such a set of good models is a hard computational problem, there have been only a few algorithms for the problem so far, most of which are either approximate or incomplete. To overcome this difficulty, we study efficient enumeration of all good models for a subclass of interpretable models, called rule lists. Based on a state-of-the-art optimal rule list learner, CORELS, proposed by Angelino *et al.* in 2017, we present an efficient enumeration algorithm *CorelsEnum* for exactly computing a set of all good models using polynomial space in input size, given a dataset and a error tolerance from an optimal model. By experiments with the COMPAS dataset on recidivism prediction, our algorithm *CorelsEnum* successfully enumerated all of several tens of thousands of good rule lists of length at most $\ell = 3$ in around 1,000 seconds, while a state-of-the-art top-K rule list learner based on Lawler's method combined with CORELS, proposed by Hara and Ishihata in 2018, found only 40 models until the timeout of 6,000 seconds. For global analysis, we conducted experiments for characterizing the Rashomon set, and observed large diversity of models in predictive multiplicity and fairness of models.

1 Introduction

In applications of machine learning models to critical decision-making tasks, such as judicial decisions and loan approvals, there have been increasing concerns about the *interpretability* of the models [8, 17]. If the decisions based on their predictions might have a significant impact on individuals, decision-makers must provide the reason of the decisions to assure users of their correctness [17]. Consequently, learning interpretable models, such as decision trees, rule sets, and rule lists, has attracted considerable attention in recent years [2, 8, 11, 13]. Because these models are expressed as combinations of simple "if-then" rules as shown in Table 1, it is easy for humans to understand and validate how the models make predictions [8].

⋆ Presently working for NTT Communications Co. (e-mail: k.mata@ntt.com)
⋆⋆ Presently working for Fujitsu Ltd. (e-mail: k.kanamori@fujitsu.com)

Table 1. An example of a pair of competing rule lists of length $\ell = 3$ with similar accuracies, 62.5% and 60.9%, for predicting two-year recidivism on the COMPAS dataset. Although two rule lists have similar accuracy (Acc), they have quite different values of discrimination measures, namely, demographic parity (DP) and equal opportunity (EO), whose definitions can be found in Sec. 4.2. They also have a large discrepancy value 0.325 (the relative Hamming distance between their prediction vectors).

Acc	DP	EO		Rule list models

0.625	0.083	0.061	R_1 :	`if juvenile-felonies>0 & current-charge-degree=Felony, then Yes`
				`else if juvenile-misdemeanors=0 & priors>3, then Yes`
				`else predict No`

0.609	0.052	0.042	R_2 :	`if sex=Male & juvenile-crimes>0, then Yes`
				`else if age=18-20 & priors=0, then Yes`
				`else predict No`

Recently, for interpretable models, there has been another concern about the situation where there exist multiple models that are approximately equally accurate by relying on different features [10,16,18]. In the seminal paper, Breiman [4] has named such phenomenon "*Rashomon effect*". By showing examples of feature importance, he explained how different models with similar accuracy can generate different explanations for prediction tasks. From this view, he argues that it is unreliable to use explanations derived from a single predictive model for the class of interpretable models such as decision trees and rule lists. By introducing the notion of *prediction multiplicities*, Marx *et al.* [16] showed how a prediction problem can show multiplicities, and how we can measure the diversity of a set of good models.

For example, we show in Table 1 a part of results of experiments in Sec. 5 on the COMPAS dataset [3] for the task of predicting two-year recidivism. The table contains a pair of competing rule lists which was found by our algorithm CorelsEnum, associated with the values of the accuracy (Acc), and two major discrimination measures, namely, *demographic parity* (DP) [5] and *equal opportunity* (EO) [12] (see Sec. 4.2). Although two rule lists have similar accuracies of 62.5% and 60.9%, respectively, they have quite different characteristics in DO and EO. Moreover, we observed that there were some rule list which was only 1% less accurate than an optimal rule list, while it made different predictions on 11% of training data from the optimal one made. This type of prediction multiplicity is called *discrepancy* [16], and will be discussed later in Sec. 5.2.

The central notion in the studies mentioned above is the *collection of good models* within a given model class $\mathcal{H}$ that have similar accuracy as an optimal model on a given dataset, which is also called a "*Rashomon set*", and has been discussed by several authors [16, 18]. Here, we assume to measure the goodness of a model h by the *empirical risk* $L(h)$ on dataset S, which is the proportion of the data that the model makes incorrect predictions. Then, the notion of

Rashomon sets is captured by the following definition, due to Fisher, Rudin, and Dominici [7]: the *Rashomon set* with error tolerance $\varepsilon > 0$ is defined as the set $\mathcal{R}_\varepsilon$ of all models h whose empirical risk $L(h)$ is at most larger than that of optimal model h_* within tolerance $\varepsilon > 0$, that is, given by:

$$\mathcal{R}_\varepsilon := \{h \in \mathcal{H} \mid L(h) \leq L(h^*) + \varepsilon\},$$

Although all models in $\mathcal{R}_\varepsilon$ achieve similar accuracy, they often differ markedly in their predictions for individual inputs and thus may have different properties [10,16,17]. Consequently, characterizing the set $\mathcal{R}_\varepsilon$ plays an important role in validating the reliability of $\mathcal{H}$ on a specific prediction problem [16].

To characterize the Rashomon set $\mathcal{R}_\varepsilon$ by existing criteria, one often needs to compute the set $\mathcal{R}_\varepsilon$ for a certain model class $\mathcal{H}$ on a given dataset. However, since $\mathcal{R}_\varepsilon$ can contain exponentially many models in the input size, exact computation of $\mathcal{R}_\varepsilon$ still remains challenging [17]. Although there are only a few existing methods for the task [11,18], they can only provide a subset of $\mathcal{R}_\varepsilon$ randomizedly or approximately. Therefore, no one has exactly computed the Rashomon set $\mathcal{R}_\varepsilon$ for the class of interpretable models on real datasets and measured the existing criteria to characterize the set $\mathcal{R}_\varepsilon$ [17].

In this paper, we focus on the class of *rule lists* [2,11], and study an exact computation of all the rule lists in the Rashomon set $\mathcal{R}_\varepsilon$. For that purpose, we extend *CORELS* [2], which is a state-of-the-art optimal rule list learner, and propose an efficient algorithm for exactly computing the set $\mathcal{R}_\varepsilon$ on a given dataset and the best-achievable empirical risk. Based on $\mathcal{R}_\varepsilon$, we then measure the following prediction multiplicity scores [16]: the *ambiguity* α_ε is the proportion of data that has at least one model with conflicting prediction from h_0, while the *discrepancy* δ_ε is the maximum proportion of data that a model can make different prediction from h_0 over all good models (see Sec. 4.2).

Our contributions are summarized as follows:

- We propose an exact algorithm CorelsEnum for computing the Rashomon set for the class of rule lists. Based on CORELS [2], our algorithm can efficiently enumerate all good rule lists with length at most K and within error tolerance ε. Unlike the previous method [11], CorelsEnum uses only polynomial working space to compute the whole set.
- By experiments on the COMPAS dataset [3],with a large value of $\varepsilon = 15\%$, our CorelsEnum successfully computed the Rashomon set $\mathcal{R}_\varepsilon$ of around 23,354 all good rule lists of length at most $\ell = 3$ in 1,000 seconds, while the previous one for top-K rule lists, CorelsLawler [11], listed only top-40 rule lists before the timeout of 6,000 seconds.
- Based on the computed Rashomon sets $\mathcal{R}_\varepsilon$, we analyzed the diversity of a set of good models in terms of *predictive multiplicity* [16] and *unfairness range* [1,6]. We found that the Rashomon set $\mathcal{R}_\varepsilon$ with small error tolerance $\varepsilon = 1\%$ had large prediction multiplicities $\alpha_\varepsilon = 29\%$ and $\delta_\varepsilon = 11\%$. For discrimination scores, we observed a trade-off between the score and the empirical risk, and the existence of a few clusters of good models with similar scores.

As consequences, our results revealed that real datasets such as COMPAS could had the large diversity of models that cannot be ignored in explanability. Thus, we need further researches for efficient methods to integrate competitive rules to apply existing model explanation methods.

1.1 Related Work

Rule models, such as decision trees, rule sets, and rule lists, are popular *interpretable models* [2, 8, 14, 17]. Among them, *rule lists* and their variants [2, 11, 20] have been widely studied from the view of global optimization. Angelino *et al.* [2] proposed an algorithm CORELS that finds a single optimal rule list that exactly minimizes the size-penalized empirical risk by branch-and-bound search. In this paper, we extended CORELS for computing the complete set of all almost-accurate rule lists using enumeration and data mining techniques [9].

Computation of the Rashomon set $\mathcal{R}_\varepsilon$ has been attracting increasing attention in recent years [17] from various perspectives, such as interpretability [7,18], predictive multiplicity [16], and fairness [1,6]. However, exact computation of $\mathcal{R}_\varepsilon$ with a small memory footprint still remains challenging [17]. Particularly, Semenova *et al.* [18] described a procedure for randomly sampling a subset of $\mathcal{R}_\varepsilon$ for decision trees of bounded size. Hara and Ishihata [11] have proposed an efficient top-K rule list learner, called CorelsLawler here, based on empirical risk using the well-known Lawler's method [11]. We remark that neither of the above methods did not achieve as goals exact computation of the whole $\mathcal{R}_\varepsilon$ and polynomial working space. In contrast, our algorithm achieved both of these requirements.

2 Preliminaries

In this section, we give basic definitions and notation, which will be necessary in the following sections. We also introduce our problem of computing the collection of all good models for a class of models. For the notions that are not found here, please consult appropriate textbooks such as [13].

2.1 Notation

For a predicate ψ, $\mathbb{I}[\psi]$ denotes the indicator of ψ; that is, $\mathbb{I}[\psi] = 1$ if ψ is true, and $\mathbb{I}[\psi] = 0$ otherwise. Throughout this paper, we consider the *binary classification problem* as our prediction problem, and assume Boolean features as in most studies on learning rule models [2, 14]. Then, the input and output domains are $\mathcal{X} = \{0,1\}^J$ and $\mathcal{Y} = \{0,1\}$, respectively, where $J \in \mathbb{N}$ is the number of features. An *example* is a tuple $(\mathbf{x}, y)$ of an input vector (or an *input*) $\mathbf{x} = (x_1, \ldots, x_J) \in \mathcal{X}$ and a prediction label (or a *label*) $y \in \mathcal{Y}$, and a *dataset* is a sequence $S = \{(\mathbf{x}_n, y_n)\}_{n=1}^N$ of N examples, where $S \in (\mathcal{X} \times \mathcal{Y})^N$. For a given *classifier*, or a *prediction model*, $h\colon \mathcal{X} \to \mathcal{Y}$ and dataset S, the *empirical risk* of h is defined as $L(h \mid S) := \frac{1}{N}\sum_{n=1}^N l(y_n, h(\mathbf{x}_n)) \in [0,1]$, where $l\colon \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_{\geq 0}$ is a *loss function* that measures the difference between the prediction $h(\mathbf{x})$ and the

true label y. In this paper, we assume the 0-1 loss $l(y, \hat{y}) = \mathbb{I}[y \neq \hat{y}]$. The *number of misclassifications* by h on S is defined as $\sharp\mathrm{Err}(h \mid S) := \sum_{n=1}^{N} l(y_n, h(\mathbf{x}_n)) \in [0..N]$. Note that the empirical risk is given by $L(h \mid S) = \frac{1}{N}\sharp\mathrm{Err}(h \mid S)$.

2.2 Rule List

In this study, we focus on the class of classifiers, called *rule lists* [2,11], defined as follows. Let $\mathcal{X} = \{0,1\}^J$ be an input domain of J Boolean features. Let $\mathcal{T}$ be a set, called a *vocabulary*, which consists of terms over a set of J *Boolean features* $x_1, \ldots, x_J$ over $\{0,1\}$. Each *term* t in $\mathcal{T}$ is a conjunction $t = (x_{i_1} \wedge \cdots \wedge x_{i_k})$ of Boolean features, and represents a Boolean assertion $t\colon \mathcal{X} \to \{0,1\}$ such that t evaluates *true* on an input vector $\mathbf{x} \in \mathcal{X}$ if $x_{i_j} = 1$ for all $1 \leq j \leq k$, and *false* otherwise. For example, ('age = 18 - 20') $\wedge$ ('sex = Male') is a term used in experiments of Sec. 5. As with previous studies [2, 14], we assume that $\mathcal{T}$ includes the constant 1 (true), and that $\mathcal{T}$ is pre-mined by frequent itemset mining algorithms (e.g., FP-growth [9] or LCM [19])

Let $\mathcal{Y}$ be a set of prediction labels. A *rule* over $\mathcal{T}$ and $\mathcal{Y}$ is a pair $(t \to y)$ of a term $t \in \mathcal{T}$ and a label $y \in \mathcal{Y}$, which corresponds to the conditional statement "if t, then y." A *rule list* of length $\ell \geq 1$ over $\mathcal{T}$ and $\mathcal{Y}$ is a tuple $d = (r_1, \ldots, r_\ell)$ of ℓ rules, where (i) $r_i = (t_i \to y_i)$ is a rule for every $1 \leq i \leq \ell$, and (ii) the last rule r_ℓ always has constant test $t_\ell = 1$, and is called the *default rule*. In Table 1, we show an example of a rule list. We denote by $\circ$ the concatenation operation for rule sequences. A rule list $d = ((t_i \to y_i))_{i=1}^{\ell}$ naturally defines a *prediction model* $h_d\colon \mathcal{X} \to \mathcal{Y}$ such that given an input $\mathbf{x}$ in $\mathcal{X}$, the *prediction* $y = h_d(\mathbf{x})$ in $\mathcal{Y}$ is computed by the code below:

```
if t_1(x) then predict y_1, else if t_2(x) then predict y_2, ...,
else if t_{ℓ-1}(x) then predict y_{ℓ-1}, else predict y_ℓ.
```

In the above code, whenever the label y_i is predicted, the condition $t_1(\mathbf{x}) = 0 \wedge \cdots \wedge t_{i-1}(\mathbf{x}) = 0$ and $t_i(\mathbf{x}) = 1$ must hold. Then, we say that $\mathbf{x}$ *falls into* the i-th rule r_i. For a given dataset $S \in (\mathcal{X} \times \mathcal{Y})^N$, regularization parameter $\lambda \geq 0$, and a set $\mathcal{T}$ of candidate terms, the task of learning a rule list is formulated as:

$$h_{d^*} = \arg\min_{h_d \in \mathcal{H}_\mathcal{T}} R_\lambda(h_d \mid S) := L(h_d \mid S) + \lambda \cdot |d|. \tag{1}$$

Although finding an optimal solution of the problem (1) is a hard combinatorial optimization, it can be efficiently solved by recent branch-and-bound optimization algorithms such as CORELS [2] in many practical instances.

2.3 Computation of Rashomon Sets

To characterize the set of good models, the *Rashomon set* has been introduced as a set of models that achieve near-optimal accuracy [17]. For a prediction problem $(\mathcal{X}, \mathcal{Y})$, let $\mathcal{H}$ be a set of classifiers $h\colon \mathcal{X} \to \mathcal{Y}$, which we call a *model class*. Following previous studies [16,18], we define the Rashomon set as a subset of classifiers that achieve accuracy close to a given *reference classifier* $h_0 \in \mathcal{H}$ with respect to a certain loss function l and a given error tolerance $\varepsilon \geq 0$.

Definition 1. *Given a model class $\mathcal{H}$, reference classifier $h_0 \in \mathcal{H}$, dataset S, and error tolerance $\varepsilon \geq 0$, the* Rashomon set $\mathcal{R}_\varepsilon(h_0 \mid S)$ *is defined as follows:*

$$\mathcal{R}_\varepsilon(h_0 \mid S) := \{h \in \mathcal{H} \mid L(h \mid S) \leq L(h_0 \mid S) + \varepsilon\}.$$

As with existing studies [16,18], we assume the reference classifier h_0 to be an optimal rule list h_{d^*} for the learning problem (1), which can be obtained using CORELS [2]. Note that the choice of h_0 is independent of our results. Now, we formally define our problem as follows:

Problem 1. Given a dataset S, a set of terms $\mathcal{T}$, a reference rule list h_0, an error tolerance $\varepsilon \geq 0$, and $\ell \geq 0$, compute the Rashomon set $\mathcal{R}_\varepsilon(h_0 \mid S)$ for the class of rule lists of length at most ℓ.

By solving Problem 1, we can obtain the Rashomon set $\mathcal{R}_\varepsilon(h_0 \mid S)$ of rule lists of length $\leq \ell$, and can analyze the properties of $\mathcal{R}_\varepsilon(h_0 \mid S)$ from various perspectives described in Sec. 4.

2.4 Optimal Rule List Learner CORELS

Our algorithm is designed based on the recent branch-and-bound optimization algorithm CORELS for learning a single optimal rule list, proposed by Angelino *et al.* [2]. Here, we will briefly review CORELS, and discuss how we can extend CORELS to exact computation of the Rashomon set.

The inputs to the CORELS algorithm are a set $\mathcal{T}$ of terms, a set $\mathcal{Y}$ of labels, a training dataset S, and numbers $\ell \geq 1$ and $\lambda > 0$. Invoked as Corels$(\mathcal{T}, \mathcal{Y}, \ell, \lambda, S)$ with input parameters, CORELS finds an optimal rule list d_* with length $\leq \ell$ that minimizes the objective $R(d_*)$ in eq. (1) by traversing the hypothesis space of prefixes of rule lists as follows. For every $1 \leq k \leq \ell$, let $d_k := r_1 \circ \cdots \circ r_k$ is called a *k-prefix*, where $r_i = (t_i \to y_i)$ and $\circ$ is the concatenation. Then, CORELS starts with the empty prefix $()$ and by recursively expanding the current $(k-1)$-prefix π to k-prefix $\pi' = \pi \circ r_k$, $0 \leq k \leq \ell$, by appending a new rule $r_k \in \mathcal{T} \times \mathcal{Y}$. The CORELS algorithm employs sophisticated pruning strategies using constraints such as maximum rule length L, and the estimate of a lower bound of the objective.

If $M_{\text{corels}} \leq |\mathcal{Y}|^\ell |\mathcal{T}|^{\ell-1}$ is the number of caldidate prefixes for CORELS to visit, CORELS runs in $t_{\text{corels}} = O(M_{\text{corels}}|S|)$ time and $s_{\text{corels}} = O(L + |S| + |\mathcal{T}|)$ space in the worst case using stack of length at most L.

3 Methods for finding good models

In this section, we study efficient methods for finding a set of good models on a given training dataset. Firstly, in Sec. 3.1, we briefly review an existing algorithm, referred to as CorelsLawler in this paper, for Top-K enumeration of good rule lists using CORELS algorithm as a black-box function, proposed by Hara and Ishihata [11]. Next, in Sec. 3.2, we propose our algorithm CorelsEnum that efficiently enumerate all the rule lists of length $\leq K$ in the Rashomon set on a given dataset.

Algorithm 1 Lawler's method with CORELS for finding Top-K rule lists with respect to prediction error (score).

Input: A set $\mathcal{T}$ of all terms, a label set $\mathcal{T}$, $\ell \geq 0$, $\lambda > 0$, and a dataset S.
Output: A list $Answers$ of top-K rule lists in prediction error.
Procedure CorelsLawler

1: $Answers \leftarrow \emptyset$
2: $(score, Rule) \leftarrow \mathsf{Corels}(\mathcal{T}, \mathcal{Y}, \ell, \lambda, S)$
3: $Queue \leftarrow \{(score, (Rule, \mathcal{T}, \emptyset))\}$ ▷ A priority queue of $(Rule, T, F)$ with $score$ as key, where $Rule$ is a rule set, T and F are include and exclude sets of features.
4: **while** $Queue \neq \emptyset$ and $|Answers| < K$ **do**
5: $(score, (Rule, T, F)) \leftarrow Queue.\texttt{deletemin}()$ ▷ An entry with minimum $score$
6: $Terms \leftarrow Rule.\texttt{Terms}()$
7: **if** $Terms \notin \mathcal{F}$ **then** ▷ $Terms$ is the set of all terms used in Rule
8: $Answers \leftarrow Answers \cup \{(score, Rule)\}$
9: $\mathcal{F} \leftarrow \mathcal{F} \cup \{Terms\}$
10: **for** each $f \in Terms$ **do**
11: $(score', Rule') \leftarrow \mathsf{Corels}((T \backslash \{f\}), \mathcal{Y}, \ell, \lambda, S)$
12: $Queue \leftarrow Queue \cup \{(score', (Rule', (T \backslash \{f\}), F \cup \{f\})\}$
13: **return** $Answers$

3.1 Lawler's method combined with Corels algorithm

Lawler's method [15] is a well-known framework for top-K enumeration using a black-box optimization function. In Algorithm 1, we show the pseudo-code for Hara and Ishihata's algorithm [11], called CorelsLawler here, for finding top-K rule lists using Lowler's method. This algorithm iteratively calls CORELS [2], to find one of the optimal rule lists within the subspace of hypothesis. During the search, It removes some terms appearing in a discovered rule list $Rule$ from $\mathcal{T}$ to efficiently search the hypothesis space of good models, where $Rule.\texttt{Terms}()$ is the set of terms appearing in $Rule$.

If K is the number of good models to output, we can show that the time and space complexity of CorelsLawler is at most $t_{\text{lawler}} = O(t_{corels} \cdot K\ell)$ time and $s_{\text{lawler}} = O(s_{\text{corels}} + K\ell)$ space. A major disadvantage of CorelsLawler is its exponential space complexity since it must keep the set $\mathcal{F}$ of all terms found so far for the membership test at Line 7. Since $|\mathcal{F}| \leq K \leq M_{\text{corels}} \leq |\mathcal{T}|^{\ell-1}|\mathcal{Y}|^{\ell}$, $|\mathcal{F}|$ becomes exponential in ℓ in the worst case.

3.2 The Proposed Algorithm CorelsEnum

By extending CORELS, we devised our algorithm CorelsEnum for computing the Rashomon set of rule lists in polynomial space in ℓ and other inputs. In *Algorithm 2*, we show the pseudo-code of the CorelsEnum algorithm. Given a vocabulary $\mathcal{T}$, a label set $\mathcal{Y}$, the maximum length parameter $\ell \geq 0$, a dataset S of N example, and the empirical risk $L(h_{d_0} \mid S)$ of a reference rule list d_0, CorelsEnum traverses the space of rule lists in depth-first manner from a shorter prefix to longer one, starting from the empty prefix ().

Algorithm 2 A basic algorithm CorelsEnum for computing the Rashomon set $\mathcal{R}_\varepsilon(h_0 \mid S)$ consisting of all rule lists h_d with length $\leq \ell$ such that $L(h_d \mid S) \leq L(h_{d_0} \mid S) + \varepsilon$, with respect to a reference rule list h_{d_0}.

Procedure CorelsEnum$(dp, k, L_*, \mathcal{T}, \mathcal{Y}, \ell, S)$:
Input: A candidate prefix $dp = (r_1, \ldots, r_k)$, its length $k \geq 0$, a non-empty set of terms $\mathcal{T}$, a label set $\mathcal{Y}$, $\ell \geq 0$, $\lambda > 0$, $L_* \in [0, 1]$, and a dataset $S \in (\mathcal{X} \times \mathcal{Y})^N$.
Output: The subset of $\mathcal{R}_\varepsilon(h_0 \mid S)$ consisting of all rule lists with prefix dp.

```
1: for label y ∈ 𝒴 do                          ▷ Step 1: Processing a rule list d with default label y
2:     d ← (dp ∘ (1 → y)); L ← L(h_d | S)
3:     if L ≤ L_* then
4:         Output (d, L) as a solution                                  ▷ A solution is found
5: if k ≥ ℓ then return                                  ▷ Pruning by the maximum length
6: for term t ∈ 𝒯 do                          ▷ Step 2: Generating children of a parent prefix dp
7:     for label y ∈ 𝒴 do
8:         dp' ← (dp ∘ (t → y))
9:         if LB(dp', S) ≤ L_* then                      ▷ Pruning by a lowerbound of L
10:            CorelsEnum(dp', k + 1, L_*, 𝒯 \ {t}, 𝒴, ℓ, S)                ▷ Recursive call
11: return
```

At each iteration with a candidate prefix $dp = (r_1, \ldots, r_k)$, $0 \leq k \leq \ell$, the algorithm either builds a rule list d from the current prefix dp, or makes branching with children $dp' = dp \circ (t \to y)$ for all possible combinations of a term t in $\mathcal{T}$ and a label y in $\mathcal{Y}$.

Invoked with as arguments $dp = ()$, k, $L_* = L(h_0 \mid S) + \varepsilon$, $\mathcal{T}$, $\mathcal{Y}$, ℓ, and S, the recursive procedure CorelsEnum computes the Rashomon set of all rule lists with length $\leq \ell$ on a dataset S at each iteration as follows:

- Receive the current candidate prefix dp of length $0 \leq k \leq \ell$ over $\mathcal{T}$.
- For each label y in $\mathcal{Y}$, test if the rule list $d = dp \circ (1 \to y)$ and its empirical risk $L = L(h_d \mid S)$ satisfies that $L < L_*$. If the test succeeds, output the pair (d, L) as a solution.
- For each $t \in \mathcal{T}$ and $y \in \mathcal{Y}$, do: First, generate the child prefix $dp' = dp \circ (t \to y)$ of length $k+1$ from dp by appending a new rule $(t \to y)$, make a recursive call with dp', and updating $\mathcal{T}'$ by removing t to avoid duplicates.

In our algorithm, we employ some pruning techniques of CORELS in a similar way to prune search of unnecessary subspaces as follows, where we attach comments to the corresponding part of *Algorithm 2*:

(1) *Pruning based on minimum support*: it asserts that each rule must capture enough number of examples for the reliability of prediction.
(2) *Pruning based on estimated lower bounds*: When invoking recursive call for a child dp', if the lower bound function LB does not satisfy $LB(dp', S) \leq L_*$, prune all computation for dp' and all of its descendants. We use the *lower bound function* $LB(d, S)$ that is same to the empirical risk $L(h_d \mid S)$ except that all data that fall in the default rule are ignored [2] as in CORELS.

(3) *Pruning based on symmetry*: If a range of consecutive rules $r_i, r_{i+1}, \ldots, r_j$ in a rule list d, $1 \leq i < j \leq k$, have the same labels $y_i = y_{i+1} = \cdots = y_j$, any permutation of them does not change the prediction by h_d. Thus, we can keep some r_σ, $i \leq \sigma \leq j$, and discard the rest of them.

In spite of the inherent difference between CORELS with branch-and-bound search and CorelsEnum with exhaustive search for all good rule lists, the above strategies (1)–(3) effectively prune the unnecessary subspaces of candidates. Let $M_{\text{enum}} \leq |\mathcal{Y}|^\ell |\mathcal{T}|^{\ell-1}$ be the number of caldidate prefixes for CorelsEnum to visit. We show the following theorem.

Theorem 1. CorelsEnum *of Algorithm 2 enumerates all good rules with length* $\leq \ell$ *on a data set* $S \in (\mathcal{X} \times \mathcal{Y})^N$ *in* $t_{\text{enum}} = O(M_{\text{enum}}|S|)$ *time and* $s_{\text{enum}} = O(|S| + |\mathcal{T}| + \ell^2)$ *space.*

Proof. The time complexity follows that CorelsEnum requires $O(|S|)$ time at each iteration to compute the objectives. The space complexity follows that the algorithm only keep at most ℓ rule lists with length $\leq \ell$ on any branch of the search tree. □

A major advantage of CorelsEnum is that CorelsEnum has the polynomial space complexity in all inputs including ℓ independent of the number of solutions $K \leq |\mathcal{T}|^{\ell-1}|\mathcal{Y}|^\ell$, while CorelsLawler requires the space proportional to K, which may be exponential in ℓ in the worst case. CorelsEnum has amortized polynomial delay complexity, that is, it lists candidates in $O(|S|)$ time per candidate. We remark that if the pruning strategy for CORELS effectively cuts candidates earlier on an input, it is possible that CorelsLawler runs much faster than CorelsEnum since the search space of the former is narrower than the latter.

4 Evaluation Criteria for Characterizing Rashomon Sets

In this section, we introduce model criteria for analyzing the Rashomon set from the views of prediction multiplicity [16], and fairness of prediction. [12].

Several useful criteria have been proposed for characterizing some properties of a certain model class, such as interpretability [7], multiplicity [16], and fairness [1, 6], through the lens of the Rashomon set. In particular, we focus on the *predictive multiplicity* and *unfairness range* described below. Note that these criteria can be easily computed once the Rashomon set $\mathcal{R}_\varepsilon(h_0 \mid S)$ is obtained.

4.1 Predictive Multiplicity

Marx *et al.* [16] have introduced the *predictive multiplicity* as the ability of a prediction problem to admit competing models that assign conflicting predictions. Given a reference classifier h_0, the predictive multiplicity is exhibited over the Rashomon set $\mathcal{R}_\varepsilon(h_0 \mid S)$ if there exists a classifier $h \in \mathcal{R}_\varepsilon(h_0 \mid S)$ such that $h(\mathbf{x}) \neq h_0(\mathbf{x})$ for some $\mathbf{x}$ in the dataset S. To measure the predictive multiplicity, *ambiguity* and *discrepancy* have been proposed [16].

Ambiguity. Ambiguity represents the number of predictions by the reference classifier h_0 that can change over the set of competing classifiers $h \in \mathcal{R}_\varepsilon(h_0 \mid S)$. Formally, the ambiguity $\alpha_\varepsilon(h_0 \mid S)$ is defined by

$$\alpha_\varepsilon(h_0 \mid S) := \frac{1}{N}\sum\nolimits_{n=1}^{N} \max\nolimits_{h \in \mathcal{R}_\varepsilon(h_0|S)} \mathbb{I}\left[h(\mathbf{x}_n) \neq h_0(\mathbf{x}_n)\right] \quad \in [0,1]. \tag{2}$$

The ambiguity $\alpha_\varepsilon(h_0 \mid S)$ reflects the number of individuals $\mathbf{x}$ who could contest their assigned prediction $h_0(\mathbf{x})$ by the deployed model h_0 since their predictions are determined depending on the model choice by the decision-makers [16].

Discrepancy. Discrepancy represents the maximum number of predictions that can change if we switch the reference classifier h_0 with a competing classifier $h \in \mathcal{R}_\varepsilon(h_0 \mid S)$. Formally, the *discrepancy* $\delta_\varepsilon(h_0 \mid S)$ is defined by

$$\delta_\varepsilon(h_0 \mid S) := \max\nolimits_{h \in \mathcal{R}_\varepsilon(h_0|S)} \mathrm{Dist}_{\mathrm{Hum}}(h, h_0 \mid S) \quad \in [0,1], \tag{3}$$

where $\mathrm{Dist}_{\mathrm{Hum}}(h, h_0 \mid S) := \frac{1}{N}\sum_{n=1}^{N} \mathbb{I}\left[h(\mathbf{x}_n) \neq h_0(\mathbf{x}_n)\right] \in [0,1]$ is the *normalized Hamming distance* between the vectors of the predictions by h and h_0. Compared to the ambiguity, the discrepancy $\delta_\varepsilon(h_0 \mid S)$ reflects the number of the conflicting predictions $h(\mathbf{x}_n) \neq h_0(\mathbf{x}_n)$ by a single competing model $h \in \mathcal{R}_\varepsilon(h_0 \mid S)$ [16].

4.2 Discrimination Scores and Unfairness Ranges

While Coston *et al.* [6] have proposed a framework that evaluate the fairness of the classifiers over the Rashomon set, Aïvodji *et al.* [1] have pointed out that the Rashomon effect corresponds to the risk of *fairwashing*, which is a malicious attack that rationales unfair complex models by interpretable and fair surrogate models [1]. By motivating these studies, we introduce the *unfairness range* to evaluate the fairness over the Rashomon set $\mathcal{R}_\varepsilon(h_0 \mid S)$.

Let $z_n \in \{0,1\}$ be a *sensitive attribute* (e.g., gender or race) with respect to the n-th example $(\mathbf{x}_n, y_n)$ in a dataset S. To evaluate the fairness of a classifier h with respect to the sensitive attribute z, we focus on *demographic parity (DP)* [5] and *equal opportunity (EO)* [12], which are major discrimination criteria based on statistical parity. The DP and EO scores of h on S are defined as: $\mathrm{DP}(h \mid S) := \hat{P}(h(\mathbf{x}) = 1 \mid z = 1) - \hat{P}(h(\mathbf{x}) = 1 \mid z = 0), \mathrm{EO}(h \mid S) := \hat{P}(h(\mathbf{x}) = 1 \mid y = 1, z = 1) - \hat{P}(h(\mathbf{x}) = 1 \mid y = 1, z = 0)$, where $\hat{P}$ is the empirical probability over the joint distribution on y, z, and $h(\mathbf{x})$ of S.

Let $D \in \{\mathrm{DP}, \mathrm{EO}\}$ be any discremination score. We introduce the *unfairness range* of the Rashomon set $\mathcal{R}_\varepsilon(h_0 \mid S)$, denoted γ_ε^D, as an approximation of the distribution of D for the models in $\mathcal{R}_\varepsilon$. Formally, the unfairness range is the interval $\gamma_\varepsilon^D(h_0 \mid S) := [\, \min_h D(h \mid S), \max_h D(h \mid S)\,] \subseteq [-1, +1]$, where h ranges over $\mathcal{R}_\varepsilon(h_0 \mid S)$. Since we can exactly compute the Rashomon set $\mathcal{R} := \mathcal{R}_\varepsilon(h_0 \mid S)$ in t_{enum} by using CorelsEnum proposed in Sec. 3.2, now we can compute the range $\gamma_\varepsilon^D(h_0 \mid S)$ in linear time in $t_{enum} + |\mathcal{R}|$ by scanning $\mathcal{R}$.

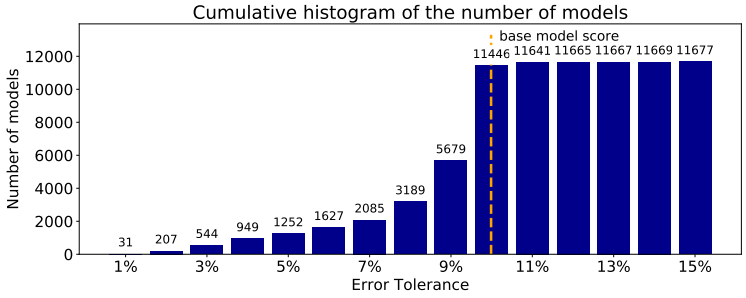


Fig. 1. Cumulative histogram of the number of the models in the Rashomon set for each value of the error tolerance ε from 1% to 15%.

Table 2. Results of execution time and the number of models found on COMPAS dataset by the existing method (CorelsLawler) and our proposed method (CorelsEnum) within around 6,000 seconds. The existing method was stopped at $K = 40$ by timeout.

	Existing method	Proposed method
Run time (s)	6021	1058
Memory(MB)	209.3	202.4
Number of models	40	23354

5 Experiments

In this section, we analyze the class of rule lists on the COMPAS dataset [3] through the lens of the Rashomon effect using our proposed algorithm.

5.1 Experimental Setting

Datasets. We used COMPAS dataset [3] for the task of criminal decision, which comprises 20 categorical attributes of individual people, relating their criminal history, with a total of $6,489$ training examples (90%) S and 721 test examples (10%) S'. The task is binary classification, where the positive category $y = 1$ indicates that the individual recidivates within two years. The sensitive attributes z represent the race of the individuals. **Programs**. We implemented CorelsLawler and CorelsEnum (Sec. 3) in Python 3.7 with `numpy` package. All the experiments were conducted on 64-bit macOS Big Sur 11.2.3 with Intel Core i9 2.4GHz CPU and 32GB Memory. We used the libraries: `pandas` for preprocessing, and `matplotlib`, `pyplot.violinplot` for charts.

Setting. Throughout this paper, we used the following setting for model parameters. A label set is $\mathcal{Y} = \{Yes, No\}$, and the maximum length of rule lists is $\ell = 3$. We used the vocabulary $\mathcal{T} \subseteq \mathcal{T}_{\text{corels}}$ of 64 terms selected from the set $\mathcal{T}_{\text{corels}}$ of all 155 terms in the github repository of CORELS [2] so that a term t is selected if and only if it evaluates true on at least half of the positive examples[1] as with previous studies [11]. Consequently, we obtain a candidate

[1] This was because we were interested in characterizing the positive category as in [11].

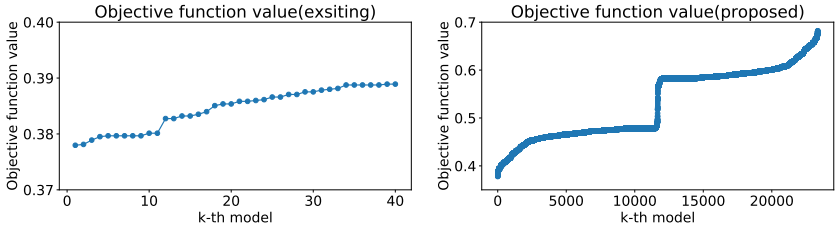


Fig. 2. The objective value (training error plus λ times the rule list length) against the rank k of a rule list on the COMPAS dataset for existing and proposed methods.

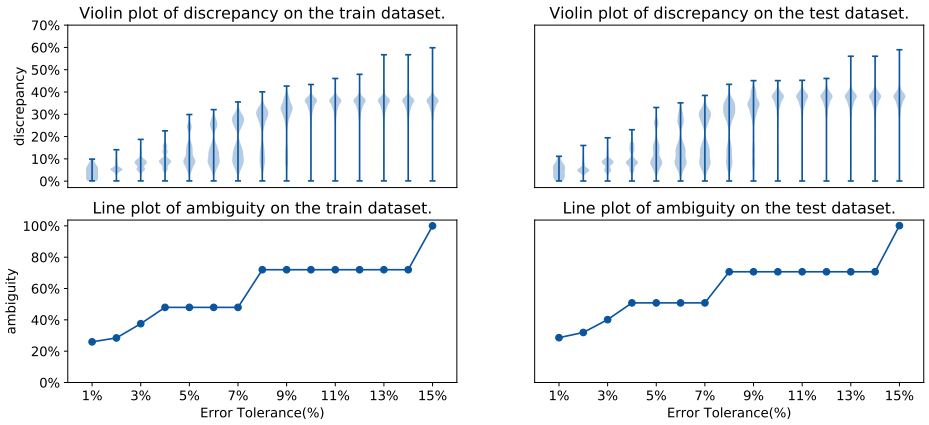


Fig. 3. Predictive multiplicity of discovered rule lists in the Rashomon set on the COMPAS dataset. The violin plots (above) show the distribution of discrepancy and the line plots (below) show the ambiguity of rule lists over the Rashomon set.

space of size $M = |\mathcal{T}|^{\ell-1}|\mathcal{Y}|^{\ell} = 64^2 2^3 = 32,798$. We first obtained a reference classifier h_0 by CORELS on the training dataset S, and then computed the Rashomon set $\mathcal{R}_\varepsilon = \mathcal{R}_\varepsilon(h_0 \mid S)$ by CorelsEnum. We computed $\mathcal{R}_\varepsilon$ by varying the error tolerance ε from 1% to 15%, and analyzed its properties for each ε.

5.2 Experimental Results

The Numbers of Good Rule Lists by Varying the Error Tolerance In Fig. 1, we show the number of the models in the Rashomon set by varying the error tolerance ε from 1% to 15%. In the figure, the reference model h_0 locates at $\varepsilon = 1\%$, which amounts to training error 34.8%, while the baseline model h_*, which is such a constant rule list that always outputs $y = 0$ for any input $\mathbf{x}$ locates at $\varepsilon = 10\%$, which amounts to the training error 44.8%, i.e., the ratio of the examples with $y = 0$ in the training dataset S. From Fig. 1, we can see that the total number of models in $\mathcal{R}_\varepsilon(h_0 \mid S)$ increases rapidly between the error tolerance ε of 9% and 10%, and almost saturates after ε exceeds 10%. For example, the Rashomon set with $\varepsilon = 9\%$ (resp. $\varepsilon = 10\%$) contained 5679

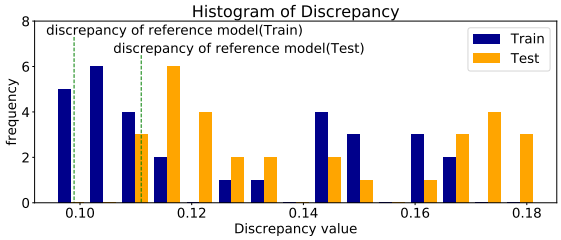


Fig. 4. The histograms of the discrepancies δ_ε of discovered rule lists in the Rashomon set with error tolerance $\varepsilon = 1\%$ on the COMPAS dataset, where the blue and yellow histograms show the frequencies in the training and test data sets, respectively.

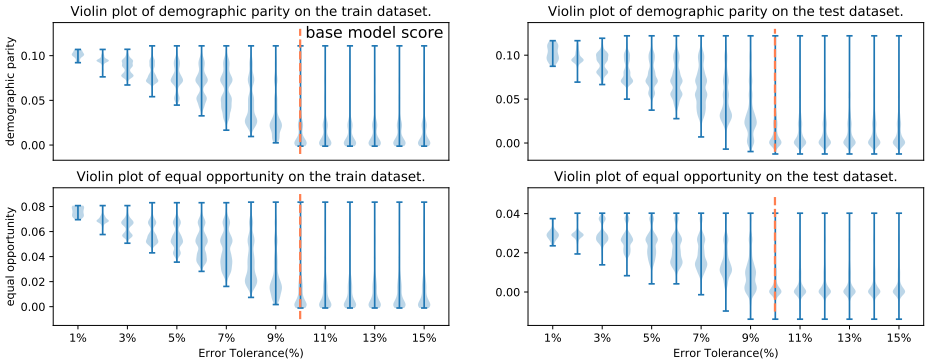


Fig. 5. Distributions of discrimination scores with respect to demographic parity (DP, upper) and equal opportunity (EO, lower) on the COMPAS dataset. Here, the violin plots show the frequencies of models with a certain score, while the error bar shows the unfairness range with DP and EO. The score for the base model is shown in red dashed lines).

(resp. 11446) rule lists. This is because the Rashomon sets with $\varepsilon \geq 10\%$ included exponentially many rule lists as accurate as the baseline model h_* in the number of candidate terms in $\mathcal{T}$.

Comparison of the Existing and the Proposed Algorithms Next, we compared the existing method (CorelsLawler) in Sec. 3.1 and our proposed method (CorelsEnum) in Sec. 3.2 in terms of running time and memory. We ran experiments for finding good rule lists in the objective function R_λ with parameter $\lambda = 0.015$ for both algorithms within around 6,000 seconds.

Table 2 shows the comparison of the running time and memory usage of both algorithms within 6,000 seconds. We see that without limit of the error tolerance ε, the proposed algorithm CorelsEnum enumerated all 23354 models including all good models for any $\varepsilon \geq 0$, while the existing method was stopped at $K = 40$ by timeout of $6,000$ seconds after finding top-40 good models. From these results, we observed that the proposed CorelsEnum was about 5.7 times faster than the existing CorelsLawler. Fig. 2 shows the objective function value against the

rank of the models. For the top-40 models, we confirmed that both algorithms successfully found models with the same value of the objective function.

Predictive Multiplicity Next, we examine the predictive multiplicity of the Rashomon set on the COMPAS dataset. Fig. 3 shows the results on the discrepancy $\delta_\varepsilon = \delta_\varepsilon(h_0 \mid S)$ and ambiguity $\alpha_\varepsilon = \alpha_\varepsilon(h_0 \mid S)$ of the Rashomon set $\mathcal{R}_\varepsilon = \mathcal{R}_\varepsilon(h_0 \mid S)$ on the training dataset S for each ε. From Fig. 3, we observed that the values of δ_ε and α_ε monotonically increased as ε increased. For example, the value of discrepancy (resp. ambiguity) with $\varepsilon = 1\%$ was $\delta_\varepsilon = 11\%$ (resp. $\alpha_\varepsilon = 29\%$). These results imply that 11% of predictions can be changed by switching the reference classifier h_0 with a classifier $h \in \mathcal{R}_\varepsilon$ that is only 1% less accurate, and that 29% of individuals are assigned conflicting predictions by at least one classifier $h \in \mathcal{R}_\varepsilon$ with the error tolerance 1% [16]. We also measured δ_ε for all good rule lists h in $\mathcal{R}_\varepsilon$. Fig. 4 shows the histogram of these values with $\varepsilon = 1\%$ on the training datasetg S and test dataset S'. From Fig. 4, we can see that there were rule lists that achieved lower discrepancy than that of the reference classifier h_0. It suggests that we can obtain another reference classifier with lower discrepancy than h_0 by exhaustive search of the Rashomon set $\mathcal{R}_\varepsilon$.

Unfairness Range Finally, Fig. 5 shows the distribution of discrimination scores demographic parity (DP) and equal opportunity (EO) on rule lists in the Rashomon set. In the figure, we can clearly see the trade-off between the empirical risk $L(h \mid S')$ and the minimum discrimination scores by the lower ends of violin plots, which is consistent with existing theoretical results [12]. For example, we have a higher discrimination value of DP = 0.10 in the higher accuracy case with error tolerance $\varepsilon = 1\%$, while we can have a lower and better value of DP = 0.02 in the lower accuracy case with error tolerance $\varepsilon = 7\%$. After error tolerance $\varepsilon \geq 10\%$ of the trivial, constant learner, we see that the distribution becomes stable, and most rule lists in the population hold the lowest DP = 0.02. Furthermore, we can see that the rule lists are concentrated to a few clusters, in the violin plots for ε from 1% to 8%, indicating existence of a few subgroups of good rule lists that behave similarly in their syntax and predictions.

6 Conclusion

In this paper, we studied efficient computation of all good models in the Rashomon set for the class of rule lists. By extending a state-of-the-art algorithm *CORELS* for a globally optimal rule list, we proposed an exact algorithm CorelsEnum for enumerating all the rule lists in the Rashomon set. To evaluate the usefulness of CorelsEnum, we conducted experiments on the COMPAS dataset, and analyzed the computed Rashomon set of the rule lists from the perspectives of predictive multiplicity and fairness.

In future work, we plan to conduct experiments on other real datasets and with larger values of $\ell \geq 4$. It is also interesting to extend our algorithm to other rule models, such as decision trees of bounded size.

Acknowledgement. The authors would like to thank anonymous referees for their valuable comments that improves the quality of this paper. This work was partly supported grants from Grant-in-Aid for JSPS Research Fellow 20J20654, and Grant-in-Aid for Scientific Research(A) 20H0059.

References

1. Aivodji, U., Arai, H., Gambs, S., Hara, S.: Characterizing the risk of fairwashing. In: Proc. NeurIPS 2021, to appear (2021)
2. Angelino, E., Larus-Stone, N., Alabi, D., Seltzer, M., Rudin, C.: Learning certifiably optimal rule lists. In: Proc. KDD 2017. p. 35–44 (2017)
3. Angwin, J., Larson, J., Mattu, S., Kirchner, L.: Machine Bias. ProPublica (2016)
4. Breiman, L.: Statistical Modeling: The Two Cultures (with comments and a rejoinder by the author). Stat. Sci. **16**(3), 199 – 231 (2001)
5. Calders, T., Kamiran, F., Pechenizkiy, M.: Building classifiers with independency constraints. In: Proc. ICDM Workshops 2009. pp. 13–18 (2009)
6. Coston, A., Rambachan, A., Chouldechova, A.: Characterizing fairness over the set of good models under selective labels. In: Proc. ICML 2021. pp. 2144–2155 (2021)
7. Fisher, A., Rudin, C., Dominici, F.: All models are wrong, but many are useful: Learning a variable's importance by studying an entire class of prediction models simultaneously. J. Mach. Learn. Res. **20**(177), 1–81 (2019)
8. Guidotti, R., Monreale, A., Ruggieri, S., Turini, F., Giannotti, F., Pedreschi, D.: A survey of methods for explaining black box models. CSUR. **51**(5), 1–42 (2018)
9. Han, J., Kamber, M., Pei, J.: Data Mining: Concepts and Techniques. Morgan Kaufmann, 3rd edn. (2011)
10. Hancox-Li, L.: Robustness in machine learning explanations: Does it matter? In: Proc. FAT* 2020. pp. 640–647 (2020)
11. Hara, S., Ishihata, M.: Approximate and exact enumeration of rule models. In: Proc. AAAI 2018. pp. 3157–3164 (2018)
12. Hardt, M., Price, E., Srebro, N.: Equality of opportunity in supervised learning. In: Proc. NeurIPS 2016. pp. 3323–3331 (2016)
13. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning. Springer Series in Statistics, Springer (2001)
14. Lakkaraju, H., Bach, S.H., Leskovec, J.: Interpretable decision sets: A joint framework for description and prediction. In: Proc. KDD 2016. pp. 1675–1684 (2016)
15. Lawler, E.L.: A procedure for computing the k best solutions to discrete optimization problems and its application to the shortest path problem. Manag. Sci. **18**(7), 401–405 (1972)
16. Marx, C., Calmon, F., Ustun, B.: Predictive multiplicity in classification. In: Proc. ICML 2020. pp. 6765–6774 (2020)
17. Rudin, C.: Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. Nat. Mach. Intell. **1**, 206–215 (2019)
18. Semenova, L., Rudin, C., Parr, R.: A study in rashomon curves and volumes: A new perspective on generalization and model simplicity in machine learning. arXiv preprint, arXiv:1908.01755 (2019)
19. Uno, T., Kiyomi, M., Arimura, H., et al.: Lcm ver. 2: Efficient mining algorithms for frequent/closed/maximal itemsets. In: Proc. FIMI 2004 (2004)
20. Wang, F., Rudin, C.: Falling Rule Lists. In: Proc. AISTATS 2015. pp. 1013–1022 (2015)

Evolutionary Numerical Workflow for Large-Scale Feature-based Remodeling and Shape Optimization

Damir Vučina, Milan Ćurković, Ivo Marinić-Kragić

[1] University of Split, FESB
R. Boskovica 32, 21000 Split, Croatia
vucina@fesb.hr

Abstract. Evolutionary parametric shape optimization relies on generic shape models which provide sufficient 3D geometric modeling freedom while being modest in terms of the number of shape variables and hence dimensionality of search space. The number of shape parameters needs to be reduced towards computational efficiency in shape optimization. However, this may imply loss of generality or local modeling capacity, potentially also bias towards unintentionally predefined 3D shape templates, all of which may result in sub-optimal shapes.
This paper develops a novel approach in the form of a conceptual intelligent system based on integral parametric surfaces representing objects. The computational efficiency of the models is established by engaging a sparing multitude of shape parameters in evolutionary optimization. Sufficiently generic and unbiased modeling capacity is obtained by the adaptability of the approach, as it can adapt to the features of the current state in optimization iterations. Decomposition of the integral surfaces into shape partitions is implemented based on 3D pattern recognition (edges, peaks, etc).

Keywords: Evolutionary Shape Optimization; Efficient 3D Parameterization; Intelligent Numerical Workflow; Adaptive Geometric Modeling, 3D feature recognition, Partitions

1 Introduction

The scope of this paper is related to 3D shape acquisition [1], geometric parameterization and evolutionary numerical optimization [2][3]. They are combined to jointly provide an intelligent system for shape-related decision-making and evolutionary optimization. The initial solution is in the form of a 3D point cloud and resulting surface mesh obtained by optical 3D scanning of an existing object, the shape of which is to be remodeled for optimized performance. Equivalently, the procedure may also start from an integral surface fitted on the outer shape envelope of the CAD model. Starting from an existing 'good' design, evolutionary shape optimization will efficiently generate better shapes under the 'guidance' of appropriate objective functions and constraints. These define the desired performance measures and given requirements for the object

in terms of autonomous intelligent shape remodeling. The resulting numerical workflow needs to provide for potentially large-scale change of shape whereby some 3D shape features such as edges may dissolve and new ones arise (genesis of new features) during the optimization quasi-time, thereby changing the geometric composition of the overall shape. Accordingly, adaptive generic shape models and intelligent optimizers are necessary.

In their capacity as databases defining geometric primitives with corresponding parameters and respective relationships and constraints, CAD (computer-aided design) models are not suited to represent heavily changing shapes. While feasible within the context of shape optimization, CAD models are frequently too complex for the purpose. Sets of geometric primitives represented by parameter sets and relationships may be an excessively 'rigid' form to represent dynamic shapes subject to dramatic shape remodeling during optimization.

The approach of this paper is based on evolutionary parametric shape optimization combined with generic shape models, potentially providing an autonomous intelligent system. The latter must provide rich 3D geometric modeling freedom and be sparing in the number of shape variables since this multitude directly maps to dimensionality of optimization space. Unfortunately, these aspirations are in mutual conflict. A smaller number of shape parameters increases the computational efficiency in optimization, but may also lead to loss of generality or reduced local modeling capacity, perhaps even bias during evolutionary shape remodeling. Ultimately, this leads to sub-optimal shapes and consequently poor decision-making. An intelligent system requires a shape model of high flexibility in terms of potentially large changes of geometry during shape optimization.

The approach employs integral parametric surfaces in modeling the shape of objects subject to optimum design. It derives its efficiency from the respective adaptability. The aspiration is to adapt to the geometric features of the current shape of the object, which dynamically changes during the course of optimization iterations. Different degrees of such adaptability are explored.

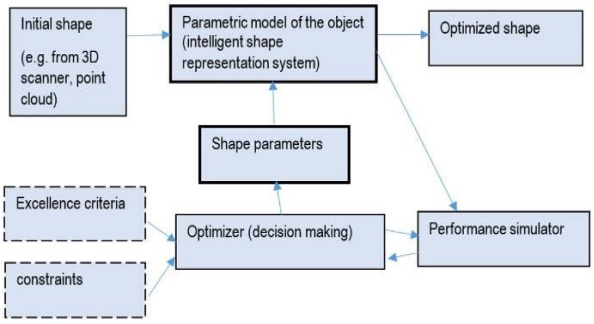


Fig. 1. Intelligent system for shape-related decision-making and evolutionary optimization

Fig.1 positions the optimizer acting on the compact set of shape parameters efficiently derived from the overall parametric model of the object in the workflow which

involves a simulator towards evaluating excellence and constraints. Integral shape parameterization, [4]-[7], may be preferable for early-stage shape representations as no numerical effort is needed related to maintaining the set of partitions and their continuity in the framework of changing shapes. Multi-partition models tend to be very demanding since the composition and topology of the partitions will generally change during remodeling imposed by the optimizer. Different mathematical models can be considered accordingly. Early-design stages imply situations where initial shapes are not just fine-tuned by the optimizer, but also major change in shape and surface configuration may take place. The effectiveness of the approach may be measured by evaluating the respective convergence rates (optimization numerical efficiency) and quality of optima (excellence values achieved) for such problems delivered by the approach presented here versus classical integral-surface geometric models.

A d^{th} degree B-spline curve with a given set of (n+1) control points $\boldsymbol{Q}_i$ is defined as

$$\mathbf{P}(t)=\sum_{i=0}^{n} N_{i,d}(t)\cdot\mathbf{Q}_i \quad , \quad t\in[0,1] \tag{1}$$

$$N_{i,0}(t)=\begin{Bmatrix}1, & t_i\le t<t_{i+1}\\ 0\,, & otherwise\end{Bmatrix} \quad , 0\le i\le n+d$$
$$N_{i,j}(t)=\frac{t-t_i}{t_{i+j}-t_i}N_{i,j-1}(t)+\frac{t_{i+j+1}-t}{t_{i+j+1}-t_{i+1}}N_{i+1,j-1}(t) \quad , 1\le j\le d\,, 0\le i\le n+d-j \tag{2}$$

with t as the position parameter. The basis functions $\boldsymbol{N}$ apply a sequence of scalars-knots t_i such that $0\le i\le n+d+1$ where $\boldsymbol{N}$ can be evaluated recursively. A 3D B-spline surface is defined for a 2D array (n_0+1)x(n_1+1) of control points $\boldsymbol{Q}$ by

$$\mathbf{P}(u,v)=\sum_{i0=0}^{n0}\sum_{i1=0}^{n1} N_{i0,d0}(u)\cdot N_{i1,d1}(v)\cdot\mathbf{Q}_{i0i1} \quad , \quad u,v\in[0,1] \tag{3}$$

where u and v are the position parameters, and d_0 and d_1 the individual degrees of the surface with $\boldsymbol{N}$ being the basis functions for the two directions respectively.

In remodeling the initial shape, it is assumed that an existing shape will be digitized into a point cloud by means of optical 3D scanning. The corresponding parametric entity representing the acquired 3D point cloud will here be obtained by least-squares based fitting of a parametric surface to the given point data-set.

Fitting a B-spline surface to a given points data-set P, [5]-[7], assumes that the given points are ordered with increasing associated parameter values. For surfaces with $u_{j0}=\frac{r_{j0}-r_0}{r_{m0}-r_0}$, $v_{j1}=\frac{s_{j1}-s_0}{s_{m1}-s_0}$ the fitting error evaluates to

$$E(\mathbf{Q})=\frac{1}{2}\sum_{j0=0}^{m0}\sum_{j1=0}^{m1}\left(\sum_{i0=0}^{n0}\sum_{i1=0}^{n1} N_{i0,d0}(u_{j0})\cdot N_{i1,d1}(v_{j1})\cdot\mathbf{Q}_{i0i1}-\mathbf{P}_{j0\,j1}\right)^2 \tag{4}$$

Best-fitting procedures minimize (4) with respect to $\boldsymbol{Q}$. As an example, a part of a boat hull was scanned into a point cloud, to which a B-spline surface was fitted (Fig.2a).

Best-fitting and parameterization may also be applied to ‘repairing’ voids in the point clouds obtained by 3D scanning as illustrated in Fig.2b. and Fig.2c.

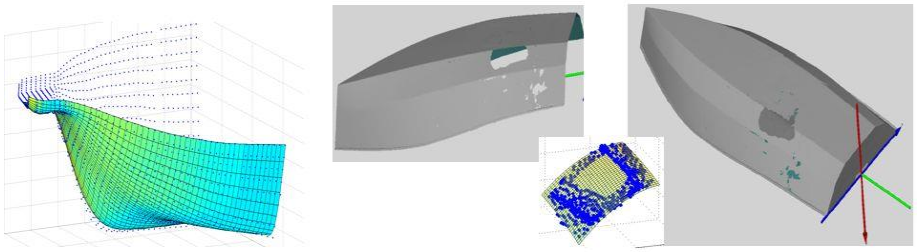


Fig. 2. Fitting to a 3D point cloud of a boat hull with a void in the point cloud acquired by optical 3D scanning and repairing the void by fitting a surface to the adjacent mesh

2 Shape-Remodeling Based on Evolutionary Optimization

In early design stages, selecting a base geometric shape for an engineering object might introduce bias and lead to sub-optimal shapes for the given objective function. Ideally, the shape of the object should be completely generic and subject to any kind of change. An initial geometry should be able not just to change its dimensions as steered by the decision-maker for the given excellence criteria and constraints. It should also be able to undergo major change in shape, potentially developing even drastic modifications both locally and globally.

Provision for such major change of shape imposes substantial difficulties on the corresponding parameterization scheme. If genetic algorithms (GA) are employed as the intelligent system to generate model improvement, the potentially different shape 'templates' or geometric primitives contained in the population all come along with different parametric sets defining them. These different parametric sets defining different individual phenotypes are coded in corresponding genotypes which consequently possess different internal structures. This paper conceptually explores options for GAs to operate on populations of potentially heterogeneous chromosomes which code different phenotypes. This should provide an augmented degree of intelligence to the shape synthesis decision-making system based on optimization.

The key issues arising with such a setting are challenging:

- What should be the coding for chromosomes for generic shapes which may consist of sets of different geometric primitives or complex parametric entities (e.g. NURBS)?
- Is it feasible and what are the options for GA operators such as crossover or mutation when operating on such heterogeneous chromosomes given their different internal compositions of genotypes?
- Is it possible for GAs to operate and generate new phenotypes by manipulating the chromosomes of different genotype compositions?

Shape models as compositions might typically involve a multitude of individual primitives (e.g. cuboids, cylinders, NURBS, etc, Fig.3) along with their respective positions, orientations and respective shape parameters as well as their mutual relationships and constraints. All this information should be coded in chromosomes, if GAs are to be employed directly with such models. It might prove virtually impossible to have the GAs

manipulate all these pieces of geometric information and generate new, yet feasible, geometric models. The respective children in a heterogeneous GA population can be expected to differ not only in values of individual parameters but also in the very composition of the overall shape model, many of them being physically inadmissible or meaningless. The approach of this conceptual paper is to proceed along a somewhat simplified path while still providing a large scope of generality in generating topologically different shapes using GAs. Such a numerical workflow as an intelligent system might potentially be capable for autonomous genesis of new conceptual solutions in terms of shape compositions.

All the different shape compositions present in some current GA population are coded by corresponding genotypes of different internal structures (e.g. different numbers of genes, coding of different properties) and hence can not be acted upon by the GA operators. Crossover or mutation operating on incompatible genotypes are infeasible. The simplified approach to bypassing the above problem is based on converting any shape composition into an integral B-spline surface which models the 'outer skin' of the shape composition. Having converted all fundamentally different shapes into the same equivalent integral B-spline surface phenotype, the common genotype may code the control points and potentially nodes in (1) and (2) and provide the basis for evolutionary operators.

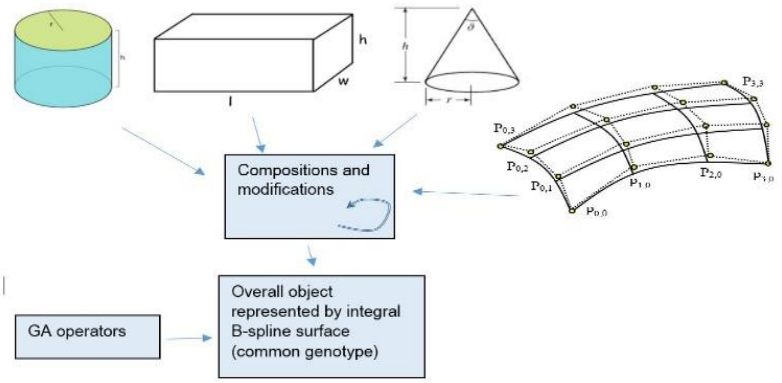


Fig. 3. Generic 3D model for shape-optimization, a symbolic population of heterogeneous phenotypes with different parameter sets to be coded by genotypes with different compositions

Accordingly, the idea is to decode the distinct genotypes into the individual phenotypes and resulting shapes, and subsequently re-coding them into chromosomes derived from the same common genotype. The latter is to be based on applying integral B-spline surfaces. Having now the entire topologically heterogeneous population coded in the same 'common denominator' form represented by B-splines, GA operators can subsequently act consistently on the now 'standardized' internal structure of chromosomes.

After the evolutionary optimization procedure has converged, or perhaps periodically during optimization iterations, it might ideally be possible to decompose the resulting shape into basic geometric primitives, if any, and remaining general NURBS surfaces. The basic primitives might include planes, cylindrical surfaces and alike. This would require feature detection functionality and subsequent partitioning. Ideally, a

CAD model consisting of a set of geometric primitives and their relationships and constraints might be derived from the optimized overall B-spline shape, as illustrated in Fig.4 which conceptually demonstrates the potential decomposition of the optimized shape. The overall surface of the object is represented as an integral B-spline surface during optimization, and the optimizer itself operates on its respective set of parameters. The genotypes and chromosomes in optimization relate to the overall 'equivalent' B-spline parametric representation of the objects regardless of the internal composition of the object (Fig.3). This enables the formulation of the common genotype for diverse compositions of geometric primitives in the overall shape of objects.

Based on feature detection, the decomposition may proceed as symbolically shown in Fig.4, where P denotes plane partitions, C denotes an elliptic cylinder or cone partition and B stands for B-spline partitions. The optimization procedure may be steered to favor compositions of basic primitive shapes for technological reasons, e.g. by introducing rewards or penalties. The resulting topology and connectivity may initially be represented by a mathematical graph or connectivity matrix (Fig.4), C_0 denoting continuity of surface and C_1 extending to continuity of respective slope. Other representations can be used, and the final model must include constraint definitions as well.

Figs. 3-4 show that the common genotype may result in different phenotypes after feature-based decomposition. This conceptual procedure enables evolutionary algorithms to operate on topologically dissimilar shape individuals in the same population.

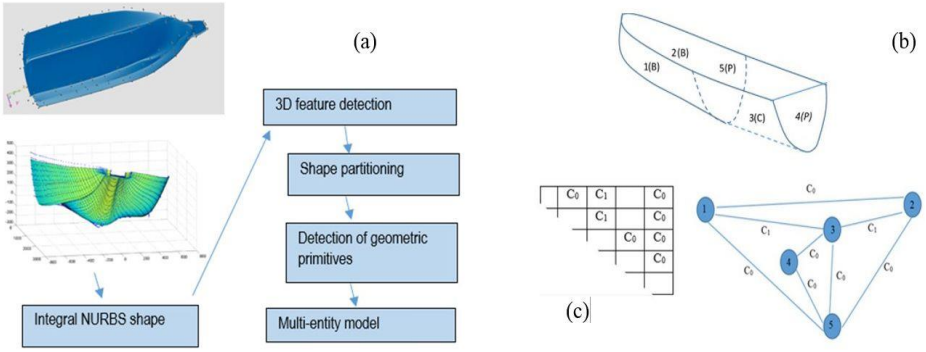


Fig. 4. Conceptual procedure, (a) partitioning of generic 3D model following the completed evolutionary shape optimization, (b) feature-based decomposition of optimized shape into partitions, (c) representation of connectivity between partitions

3 Conceptual Examples

In order to conceptually illustrate the above procedure, an elementary numerical test was developed, Fig.5. For simplicity, the shape does not consist of a composition of geometric primitives, but rather of a single one. The transition of the shape is to be conducted by GA-based shape optimization.

Numerical test 1: starting from a sphere as the initial shape, optimization should lead the shape towards becoming a cube. The fitness function expected to accomplish this shape transition is maximization of volume of the object subject to dimensional constraints in all three spatial dimensions.

Numerical test 2: starting from a cuboid as the initial shape, shape optimization should lead the shape towards becoming a sphere. The fitness function expected to drive this transition is minimization of surface area of the object subject to enclosing a given volume of the object.

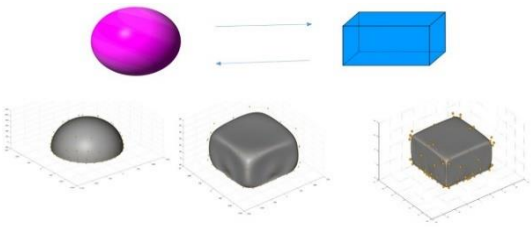


Fig. 5. Simple test of substantial shape transition, sphere, cuboid and some intermediate shapes

The first test case is implemented by modeling a general shape, a single B-spline surface. The initial shape is a sphere approximated by a B-spline surface. The B-spline surface model is controlled by an evolutionary optimizer acting as a shape-related decision maker which manipulates the respective control points (CPs). The CPs are design variables such that they can move freely in all three directions, with the exception of those located at z=0 which can only move in the x and y directions. The volume is calculated by triangulation of the B-spline surface and application of the Gauss theorem. Additionally, a gradient-based optimizer was applied with numerical gradients with first-order schemes. The objective function employed for the shape transition

$$f = -V + K \cdot \sum_i \max\left(g_i, 0\right)^2 \tag{5}$$

where the penalty term K imposes the dimensional requirements given in the form of general inequality constraints $g_i \leq 0$, and where V denotes the enclosed volume. The initial and final solutions are presented in Fig.5.

The overall procedure is rather sensitive with respect to the penalty and optimization method. Once the shape optimization procedure has converged, feature-based shape decomposition or re-composition might be applied to detect partitions. This would lead to a changed, customized, shape parameterization scheme specific for the sphere or cube respectively. Generally, this might lead to further improvement of fitness values. While geometric fitness functions were here selected for direct visualization, any other involving stresses or similar may generally also lead to dramatic shape transitions, where re-parameterization on a different basis and hence a different internal chromosome structure may become appropriate.

The example in Fig.6 presents shape optimization of a vertical-axis wind turbine, VAWT, where a drastic geometric change was induced by changing the simulation conditions in the optimization workflow. It shows that, given a Darrieus-type rotor geometry as the initial solution, an optimizer can autonomously accomplish a dramatic transformation of the respective shape into becoming a Savonius-type rotor and vice-versa. This major change of geometry (Fig.6b) is accomplished autonomously by the numerical workflow in Fig.6a. provided that the wind conditions are changed accordingly.

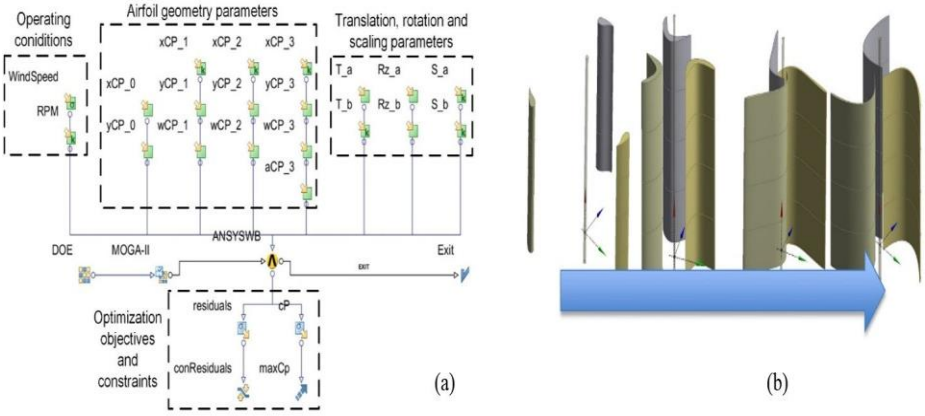


Fig. 6. (a) Numerical optimization workflow for VAWT with geometric modeling functionality, (b) Large-scale shape transformation of Darrieus-type geometry into Savonius-type rotor accomplished by the optimizer for changed operating conditions (WindSpeed, RPM)

In the numerical workflow in Fig.6a, the parameters set *CP_* denotes airfoil shape parameters- control points of the parametric surface, the geometric operators set T_*, Rz_*, S_* denote transformation matrix operators for translation, rotation, and scaling respectively applied to the control points. The symbol DOE generates the initial population submitted to the MOGA-II optimizer aimed at maximizing the power coefficient C_p, [8], whereby ANSYSWB symbol denotes the fluid flow simulator with residuals relating to the numerical solution of flow equations. The precondition for this large-scale shape transition in Fig.6b is sufficient generality of the geometric model acted upon by the optimizer, which is here assured by the parameters set and operators set in Fig.6a.

This example is somewhat simpler than the general approach in Fig. 3 as here the entire GA population is modeled by the same genotypes coding B-spline surfaces. In the particular layout in Fig.6a, the GA optimizer varies the shape via manipulating the control points set and operators set in maximizing the power coefficient, Generally, in Fig.6a and other shape optimization workflows applied here, different GA settings related to populations, selection, crossover, mutations, were successful.

4 Numerical Examples

As an example, Fig.7 shows an integral B-spline surface with some local zones which could potentially be recognized as edges, eventually subdividing the integral surface into partitions. These potential edges are 'dilluted' and not strongly expressed such that this example may serve in testing the performance of the proposed approach to feature detection in surface meshes. Examples with more and less strongly expressed features will be used. The surface is based on a given set of 7x6 control points with 2nd degree B-splines in both directions.

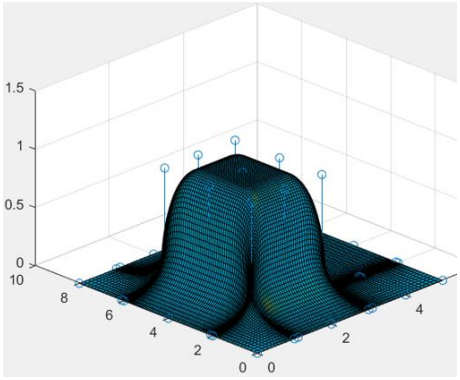


Fig. 7. Integral B-spline surface example, varying radii of curvature

The next step is to detect localities on the mesh which potentially belong to edges, corner peaks, or other specific surface shapes. There are a number of methods that can be deployed towards such feature detection, commonly based on gradients and curvatures of the mesh and their respective change and distribution. Fig. 8 is derived from applying principal component analysis to the local point set. In fact, a local point is declared to belong to an edge depending on the ratios of the eigenvalues of the covariance matrix of the local point set of the respective neighborhood.

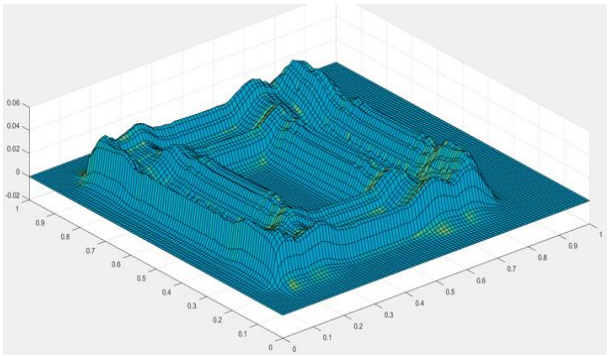


Fig. 8. Distribution of ratio (minimum eigenvalue)/(average eigenvalue) for local point neighborhoods in the parametric domain for the surface in Fig.7

At this point, partitioning the overall surface from Fig.7 based on edge identification following from Fig.8 requires a procedure to detect the individual closed regions in terms of associated points (sub-sets of the original data-set). Again, several algorithms may be used.

The procedure resulting in Fig.9 is based on the watershed algorithm once the points set has been transformed using the distance transform with inversion. The distance transform associates to each zero-level point its distance to the closest non-zero point (edge) thereby providing local minima in each region as seed points for the watershed procedure, according to:

- convert Fig.8 to binary grid data by applying threshold, obtain basins
- apply distance transform and seed local minima in regions
- apply watershed algorithm to basins

Fig. 9. Watershed transform applied to Figs.7-8, identification of individual regions

Two further procedures are needed: identification of topology of the set of regions (adjacency, connectivity type) substituting the integral surface, and type of geometric primitives modeling individual regions. In terms of the configuration of the set of partitions, the approach here is to generate a mathematical graph where the vertices denote partitions and the edges define respective connectivity. Fig. 10 shows the pseudo-code of the approach developed here, and Fig.11 shows the resulting graph.

```
for each region
    for each of seed points in the region
        for each search direction across grid
            walk along direction until maximum search distance or end of grid is reached
                for each new point during walk determine whether it belongs to
                    another region
                        if yes, chech whether regions graph already contains edge
                            between region of seed point and encountered new
                            region during walk
                        if not, add edge in regions graph between region of seed
                            point and encountered new region during walk
```

Fig. 10. Pseudo-code towards generating the adjacency matrix of regions

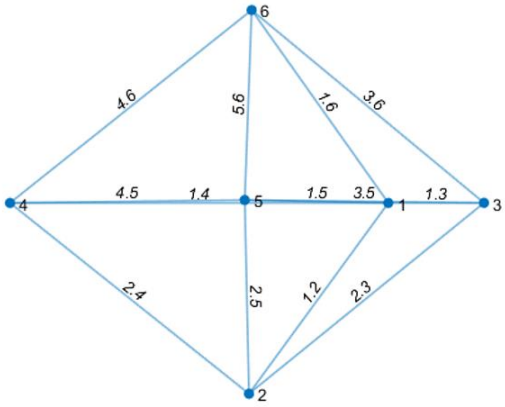


Fig. 11. Configuration graph obtained by the procedure for the given set of regions

Now, with the individual zones detected in the parametric domain, the corresponding sub-sets of the original 3Dpoints dataset can be extracted accordingly. The individual geometric primitives as partitions to replace the corresponding portions of the original points data-set (surface) can now be determined by fitting such primitives to the 3D point subsets. This fitting can be implemented in a sequence of steps, for example starting with planes, and following by cylindrical surfaces, spherical surfaces, etc.

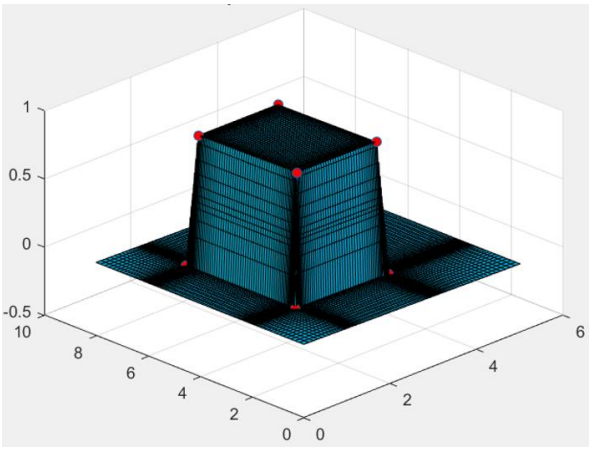


Fig. 12. Resulting partitioned ‘substitute surface’ with planar faces representing integral surface in Fig.7

Obviously, identifying a potential partitioning of the mesh in Fig.7 into a composition of primitives according to Fig.12 leads to a complete reparameterization of the geometric model accordingly, as the new parameters define the primitives and their mutual relationships, which in turn may have a major impact on shape optimization.

5 Conclusion and Future Work

A numerical workflow consisting of an evolutionary optimizer and a parametric 3D shape modeler jointly represent an autonomous intelligent system for large-scale 3D geometric remodeling based on a given set of objectives and constraints. Ongoing and future work include other elements indicated in Figs. 3-4 such as multi-partition models and the reverse procedure involving shape feature detection and resulting geometric decomposition. The proposed model enables higher generality of shape optimization than with parametric CAD-based optimization approaches as no 'close' initial shape is needed and far more remodeling freedom is provided for, hence making it suitable for early design stages.

References

1. Barbero, B.R., Ureta, E.S.: Comparative study of different digitization techniques and their accuracy. Computer Aided Design 43(2), 188–206 (2010).
2. Deb, K., Goel, T.: Multi-Objective Evolutionary Algorithms for Engineering Shape Design. KanGAL report 200003, Indian Institute of Technology (2000).
3. Rao, S.S.: Engineering Optimization. Wiley Interscience, New York (2013).
4. Farin, G.: Curves and Surfaces for Computer Aided Geometric Design: A Practical Guide. Morgan Kaufmann Publishers/Academic Press, San Francisco/London (2002).
5. Mencl, R., Muller, H.: Interpolation and approximation of surfaces from threedimensional scattered data points. In: Proc. Eurographics 98 Lisbon, 223–233 (1998).
6. Piegl, L.A., Tiller, W.: Parameterization for surface fitting in reverse engineering. Computer-Aided Design 33, 593-603 (2001).
7. Eberly, D.: Least-Squares Fitting of Data with B-Spline Surfaces. https://www.geometrictools.com/ Documentation.html, last accessed 2017/10/11.
8. modeFRONTIER software, http://www.esteco.com/modefrontier

Acknowledgments: Croatian Science Foundation grant IP-2018-01-6774

Detection of Animated Scenes Among Movie Trailers

Irmak Türköz[1] and H. Altay Güvenir[2]

[1] Bilkent University, Dept. of Computer Engineering, Ankara, Turkey
irmak.trkz@gmail.com
iturkoz.wixsite.com/personal
[2] Bilkent University, Dept. of Computer Engineering, Ankara, Turkey
guvenir@cs.bilkent.edu.tr
cs.bilkent.edu.tr/~guvenir

Abstract. This paper presents a method for the detection of animated scenes in movie trailers. Regardless of the studio, artists, and the unique features of diverse animation creation techniques, machine learning tools can provide concise detection methods of animated scenes in movie components. A dataset is prepared by selecting scenes from trailers using shot boundary analysis and removal of scenes containing non-movie contents; e.g. credentials. The dataset is composed of over 1400 movie scenes from 230 trailers of various genres. A Convolutional Neural Network (CNN) architecture followed by Gated Recurrent Unit (GRU) is built by using transfer learning of EfficientNet.

Keywords: Movie Trailer Labeling · Scene Understanding · Deep Learning · Computer Vision.

1 Introduction

Animation refers to a technique of modeling to create an illusion of a photographic event used in cartoons whereas animated movies refer to a sequence of drawings. Fortunately, the wide range of techniques used to create animated movies have been supported by the availability of larger data, more powerful computers, and accessibility to high-level photo-realistic animation creation tools boosted by machine learning.

The problem of distinguishing cartoons from photo-realistic movies can be viewed as a case study of automatic video genre classification as an immensely researched and yet to be improved subject of computer understanding. Moreover, an animation discriminator can be a tool to serve as a discriminator in the Generative Adversarial Networks (GAN), as some studies focus on generating cartoon sequences from photographic sequences. Thanks to improved cartoon discriminators, more realistic cartoons named Computer Generated Images(CGI) can be created. For instance, Chen et al. designed a GAN system to create high-quality cartoon style images from photo-realistic images [4].

Latest technologies in Computer Generated Images (CGI) have been creating more realistic animation styles that become challenging to distinguish by humans. An animation discrimination system can be further enhanced into a system to distinguish even the most photo-realistic yet computer-created movies. As Farid et al. stated, advancements in these realistic computer-generated releases blur the line between reality and fantasy, and they can even cause legal situations, such as engaging in sexually explicit conduct [5].

In this research, we have investigated if a deep learning model can distinguish every animated scene from non-animated ones, despite the diverse cartoon styles created by numerous artists. Our approach is based on training a neural network architecture trained by the scenes from trailers of the animated movie genre and other genres. To construct our dataset, we have employed a part of the MovieLens20M dataset which maps movie genres to YouTube trailers [7]. A trailer is divided into scenes using a shot boundary detection algorithm by automatically splitting the video into separate clips [1]. The term segment is used interchangeably with scene in the rest of the paper.

Once we have extracted the shot boundaries of these trailers, we eliminated segments containing misleading information such as logo and credentials. After the preprocessing is completed, 1447 movie segments exists in the dataset. In order to learn a model using the deep learning techniques, we constructed an architecture containing a Convolutional Neural Network (CNN) followed by a Gated Recurrent Units (GRU). The data set is fed into the deep neural network architecture incorporated CNN-GRU model. The model learned by the system achieved promising results compared to the state-of-the-art research.

In the next section, we will give a brief overview of the related work. Section 2 will describe the dataset used in the experiments. Section 3 introduces the methodology of constructing the model for detecting the animation scenes. The last section concludes the paper and gives directions for future work.

1.1 Related Work

One of the first attempts to classify movie segments as either animated or non-animated is by Roach et al. for the classification of video fragments using motion only [13]. Their dataset consisted of only a few sequences, 8 cartoon sequences simplified as sketched cartoons and animated cartoons, and 20 non-cartoon sequences where they have also classified non-cartoon genre as computer-generated cartoons, sci-fi, and real-life motions to simplify the problem.

A hand-crafted method was suggested by Ianeva et al. [8] with accounting pattern spectrum of parabolic size distributions to map color histograms, texture, color edges, brightness, and various image analysis filters, and trained with Probabilistic Gaussian Mixture Model and SVM. Even though they have accomplished a satisfying accuracy with their trained key-frames, their model suffered from external key-frames with divergent visual content characteristics.

Eventually, Glasberg et al. suggested a method based on MPEG-7 features extracted from 100 representative video sequences of cartoons, and commercials with animated cartoon sequences [6] which accomplished to classify of a sequence

of 50 frames in one minute with 0.8 accuracy of animated scenes and 0.85 of non-animated scenes.

To distinguish not only cartoons created by the human hand but also computer graphics images, from natural photographs Ng et al. suggested a method with accounting wavelet, geometry, and cartoon features of an image with 3200 images belonging to 4 different classes and achieved 0.84 accuracy [11]. Color histograms and SVM were considered to classify computer graphics images from photographic images by Chen et al. which proved that HSV color channels are more informative than RGB color channels with slightly better accuracy and true positive rate [3]. To extract video level features, Chen et al. used color component and color kind based on region segmentation and proved that SVM is better performing than GMM, and Manifold Ranking [2]. They have also considered cartoon segmentation as their features, and they have extracted their frames from the movies with a seemingly primitive shot boundary analyzer as a change detector.

On the other hand, Sankar et al. also considered hybrid images consisting of both cartoon and photo-realistic segments of 557 features which can be reduced 80 without a crucial loss in performance [14]. Moreover, Ionescu et al. worked with two sets of features namely temporal descriptors, like rhythm or action, and color descriptors are determined using color perception to achieve an average global correct classification up to 0.92 [9].

Initial attempts to use neural networks to classify cartoons as movie genre was presented by Montagnuolo et al. with accounting texture information, color histograms and temporal activity information based on the displaced frame difference which acquired average precision of 0.86 and an average recall of 0.85 [10]. Alternatively, Quan et al. accompanied CNN with four convolutional blocks followed by two fully connected layers to achieve 0.94 accuracy on their dataset of images with two labels: natural and computer-generated which did not include human-drawn cartoons [12]. More recently, Zhang et al. attempted to classify computer-generated images from RGB color channel and pixel correlation to acquire 0.94 accuracy on the ScNet dataset with a convolutional neural network architecture [16].

Most of the related work in this area either considered images or motion separately. However, we have assembled a dataset and built a fused model that combines both spatial information using CNN and temporal information using GRU to attempt an improved performance on accuracy.

2 Dataset

We consider one of our contributions to the research is the construction of a dataset composed of animated and non-animated movie trailers since there is not a proper dataset for this specific task as we have observed. As preliminary information, our dataset contains movie trailers with a 30 fps (frame per second) frame rate. Since we assume that a trailer may contain both animated scenes and photo-realistic scenes, we initially split each scene of a trailer by shot boundary

Fig. 1: Eliminated Image Examples

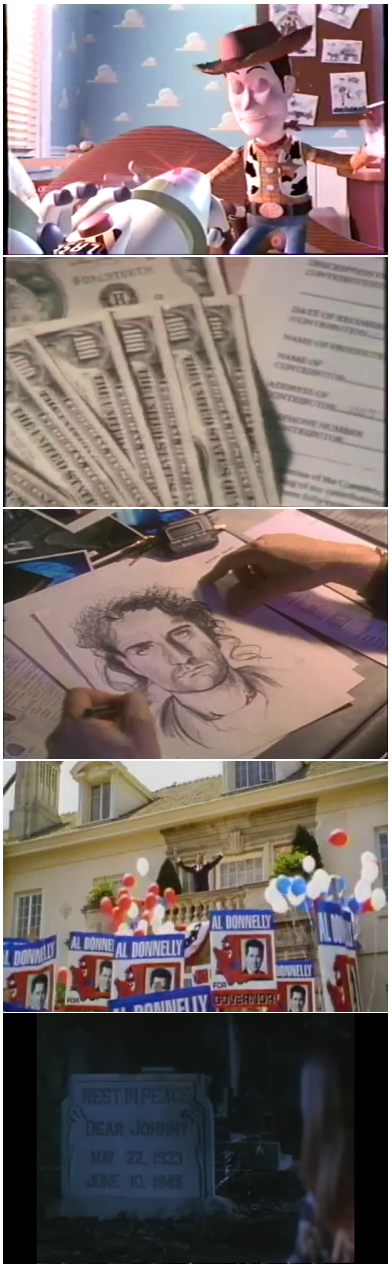


Fig. 2: Kept Image Examples

analysis. In this implementation, we refer to the movie trailer partitions generated by a scene detector as scenes. Each of these scenes is represented by a

pair of frame indices, one for the beginning and the other for the end frame. For example, since every trailer has 30 frames per second, a pair of (360,600) refers to the scene which is a segment of the trailer between 12th and 30th seconds.

The preparation of the dataset is followed by a several steps: (1) The initial links from YouTube trailers are extracted from MovieLens20M with some of the links that are unavailable due to deletion from YouTube [7]. (2) Downloaded trailers are fed into the shot-boundary detection unit of PySceneDetect's[1], ContentDetector with parameters; threshold as 40.0 and minimum scene length as 1, which allows us to discriminate fade-in and fade-out in a faster way [1]. In this step, we have also discovered that black and white trailers were not partitioned due to the implementation procedure of the open-source library. However, since black and white movies are not produced anymore and upcoming animated movies are presumably colored, we have not addressed this issue and removed non-animated black-white trailers from our dataset. (3) The output of shot-boundary detection is later filtered with the removal of scenes with a length less than 100 frames (approximately 3 seconds) and more than 1800 frames (approximately 30 seconds). This allows the later detection algorithm to be more reliable and robust since it is a motion sensitive. (4) The downloaded trailers are pre-processed with our model to eliminate movie trailer segments that are not meaningful such as credential scenes, studio logos, release dates, etc. (see Fig. 1) with a model trained to discriminate the unnecessary components.

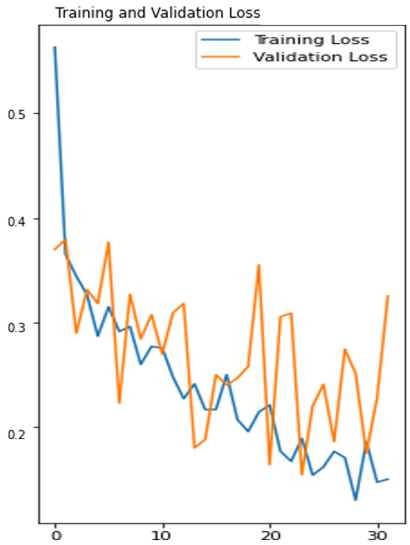


Fig. 3: Training and Validation Loss

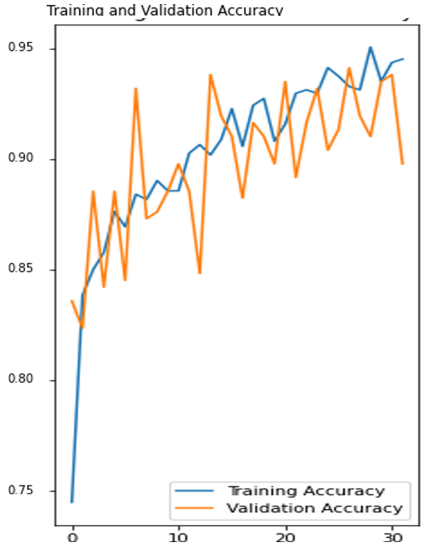


Fig. 4: Training and Validation Acc

[1] http://scenedetect.com/en/latest/

The scene detector allows trailers to be divided into intervals, mapped by the pairs of indices which indicates the beginning and the ending frame of a scene. However, for the fourth step, we consider image level or frame-level information of the scene intervals and we locate the frame in the middle of these intervals.

To eliminate unwanted intervals, we first created an image dataset composed of two classes; scenes to be removed and scenes to keep. Although the system could be created with only one class of scenes to be eliminated; we chose frames from the movies that might incorporate certain writings such as a closed up scene to a newspaper, or a signboard in the background to not eliminate noteworthy in-movie scenes (see Fig. 2). Subsequently, a CNN ResNet with 4 sequential residual blocks of 16, 32, 64 and 128 were used to train the model and acquired 0.9412 binary accuracy on the test data (see Fig. 4). As it can be seen from 3, validation loss decrease rate was significantly reduced after around 30th epoch. Although over-fitting after 30th epoch may seem as fast learning, this task can be considered simple enough to be learned in 30 epochs.

After the model is saved, every scene interval is passed through the model to either keep or remove. Lastly, black-and-white movie trailers are removed from the system and the dataset ended up with 1447 scenes in total that belongs to 234 video trailers. Half of these trailers are labeled as cartoon and the other half includes any other genre that is located in MovieLens20m. 117 movie trailers labeled as adventure, comedy, fantasy, children, romance, drama, action, crime, thriller, horror, mystery, sci-fi, documentary and musical genres are all labelled as non-animated movies. We labeled all of the trailer segments as their ancestors trailers (which they are partitioned from).

3 Methodology

Video processing can be considered as a tedious problem by computer engineers. Every frame is represented by a large number of pixels, which is the product of the height and the with of an image. Further, it is an even slower process considering that every second of a video is represented by 30 frames. For our model, we have prepared our own data generator in which we have optimized the video processing speed during learning period with a library called Decord[2]. The generator generates features for the model with dimensions of number of frames, image width, image height, number of channels (RedGreenBlue or RGB).

Since we split each trailer into several scenes, we considered each of the scenes as a different batch of the same genre. For each scene, we have extracted a set of sequential frames to be used in our model. There are various ways to implement the frame extraction method, however, we used division of each scene to a constant number of frames so that we can record the motion of scenes of different length. Our method involves a simple partitioning of the scene to an equal number of the frames explained in equation 3.

[2] https://github.com/dmlc/decord

$$par_{start} = \frac{scene_{end} + scene_{start} - frame\_rate^2}{2} \quad (1)$$

$$par_{end} = \frac{scene_{end} + scene_{start} + frame\_rate^2}{2} \quad (2)$$

$$frame\_list = (X_{par_{start}}, X_{par_{start}+frame\_rate}, \ldots, X_{par_{end}}) \quad s \quad (3)$$

$scene_{start}$: scene start frame index
$scene_{end}$: scene end frame index
par_{start} : partition start frame index
par_{end} : partition end frame index
$frame\_rate$: how many frames will be produced (given by user)
$frame\_list$: the frame list we end up with

In this equation, *partition start* refers to the starting point of the partition, which is calculated by the scene end point and the scene start points for each of the scene. This equation also allows us to partition the middle section of the scene with an optimum frame rate to reduce any misinformation frames caused by scene changes, such as fade-in or fade-outs. Partition end refers to the where the partition ends, and these frames were chosen with equal distances of frame rate. Frame rate corresponds to the how many frames will be extracted with this algorithm as an end result.

It is crucial to choose how many frames will be fed into the network since the motion is measured by the change between frames of the same scene. Our system allowed us to work with only even numbers because we divided each scene according to equation 3. Thus, we have experimented with a 2, 4, 6, and 8 for frame rates, and concluded that 2 is too short to give information about motion and 8 is too long that the difference between frames did not change much. Thus, a frame rate of 4 is found to be sufficient for capturing the motion and there is a noticeable change between the frames of the same scene.

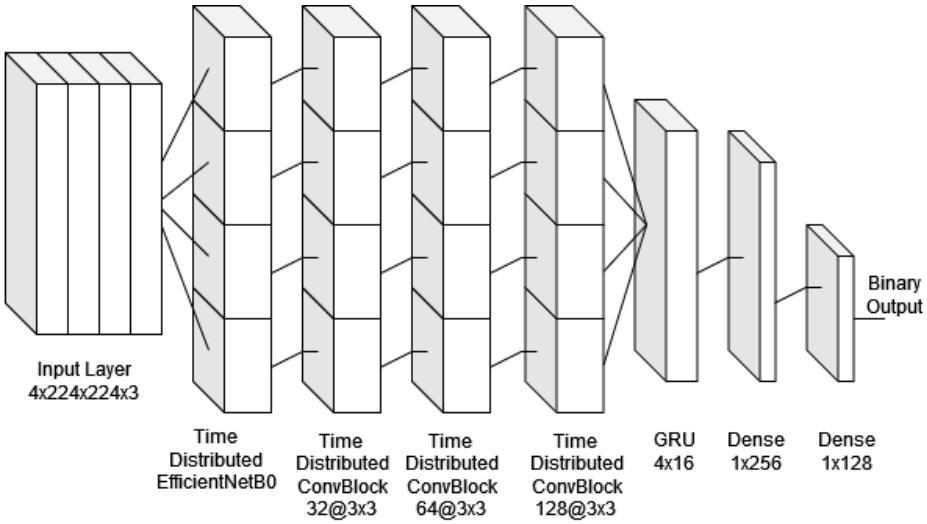


Fig. 5: Fused Model Architecture

To capture both spatial and temporal information, we have used 3 blocks of time distributed CNN with 32, 64 and 128 filters of 3 by 3; followed by a GRU of 16 units (see Fig. 5). We have also experimented with higher number of GRU units which resulted in poor results. To prevent fast over-fitting, we have accounted l1 and l2 kernel regularizers, and we included a batch normalization layer with momentum 0.9 after each convolutional block and a dropout layer. For hidden units, we have used relu activation. For the final Dense layer, we have used sigmoid since we give only one output which calculates the probability of the scene as cartoon with values close to 1.0 or photo-realistic with values close to 0.0. We have used Adam Optimizer with 0.0001 rate, loss function as binary crossentropy and metric of evaluation as binary accuracy.

To accelerate the learning process, we used EfficientNet architectures, which is proved to be very shallow and fast, yet achieves high top-1-accuracy on the ImageNet dataset and a state-of-the-art accuracy on 5 other transfer learning datasets, with an order of magnitude fewer feature size [15]. The main reason to use this transfer learning network is being explanatory for frames and efficient enough to execute in a Time Distributed fashion, since we need to apply it to every time frame extracted from a scene. We have also considered a few other transfer learning techniques, such as Residual Network (ResNet) and Very Deep Convolution Network (VGGNet). However, these models are not meant to do a profound search for objects in animated scenes and they demonstrated fast over-fitting learning due to their depth, without providing significant improvement on the validation set. Thus, fewer convolution layers are found to be helpful to discriminate cartoons from photo-realistic scenes.

4 Results and Evaluation

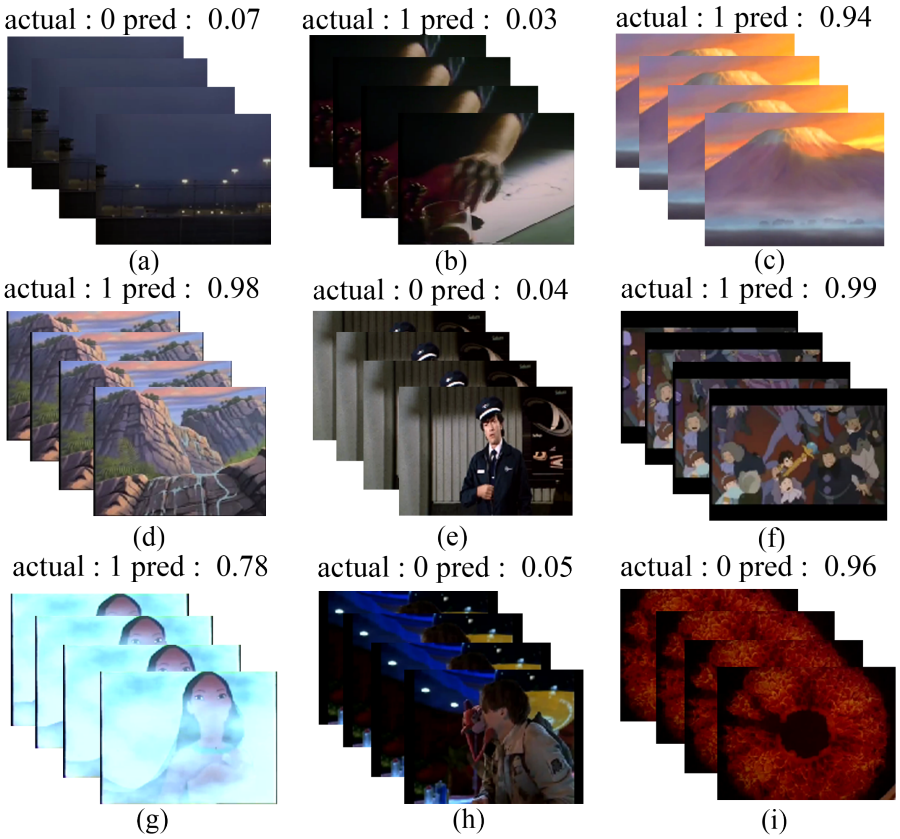


Fig. 6: System Prediction Results

In the training step, 1258 scenes extracted from 200 trailers are used to train and 189 scenes extracted from 34 trailers are used to validate the system. To prevent the testing of the model from a trailer scene that is already in the training data, trailers are separated as training and validation beforehand.

Some prediction examples of our final model is given in Fig. 6. We have labeled the animated scenes as label 1 and the non-animated ones as 0. The prediction result of a scene is an approximation of the probability of being labeled as animation. Our architecture is successful in detecting complex and crowded scenes such as Fig. 6.f. Despite low background information given in some photo-realistic scenes, the system could distinguish cartoon characters from photographic humans as it can be seen in Fig. 6.e.

On the other hand, there might be both animated scenes and photographic ones in the same trailer, and since we have labeled each scene of a trailer as the same label as the trailer, the labels were not provided to the system concisely. Consider the example given in validation set in Fig. 6.b, the scene that the trailer belongs to is a cartoon, but the scene itself is photographic. There are some scenes in the photo-graphic movie trailers that can resemble an animated scene like in Fig. 6.i. On the other hand, our model is successful in discriminating photo-graphic landscape in Fig. 6. a vs animated landscapes in Fig. 6.c-d, and could classify foggy scenes such as Fig. 6.g. Despite the challenging scenes we have given as examples, our model acquired **0.9552 binary accuracy** on our test set in 12 epochs which can be considered an outstanding result compared to the state-of-the-art. After 12 epochs, the over-fitting was observed and validation loss increased, although the training loss kept declining.

5 Conclusion

In conclusion, we have proposed a scene classification method of trailers with labels of either cartoon or photographic. We first built our dataset on top of the open-source dataset MovieLens20M, and segmented each trailer into scenes with shot-boundary detection. After pre-processing each scene with the elimination of inconsequential scenes and identifying several frames, we have trained our model involving CNN followed by GRU units and used transfer learning to advance the training process.

As our future work, we aim to increase the size of our dataset and evaluate the results of our model on other datasets. We also plan to use other transfer learning techniques and other video reading techniques to increase the efficiency of working on video data, since the training takes a long time. Moreover, we plan to use audio features and experiment if these features can further improve the reliability of our system.

Acknowledgements

We would like to express our special thanks to Dr. Sami Arpa, the founder of Largo Films company, for valuable discussions in the research and for sharing his knowledge about movie industry. [1]

References

1. Castellano, B.: Pyscenedetect. http://github.com/Breakthrough/PySceneDetect (07 2012)
2. Chen, F., Lai, M., Ye, Z.: Detecting cartoons: Automatic video genre classification. In: 2010 International Conference on Management and Service Science. pp. 1–4 (2010). https://doi.org/10.1109/ICMSS.2010.5576551

[1] Largo Films SA. EPFL Innovation Park, Building I. 1015, Lausanne/ SWITZERLAND. info [at] largofilms.ch

3. Chen, W., Shi, Y., Xuan, G.: Identifying computer graphics using hsv color model and statistical moments of characteristic functions. pp. 1123 – 1126 (08 2007). https://doi.org/10.1109/ICME.2007.4284852
4. Chen, Y., Lai, Y.K., Liu, Y.J.: Cartoongan: Generative adversarial networks for photo cartoonization. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 9465–9474 (2018). https://doi.org/10.1109/CVPR.2018.00986
5. Farid, H., Bravo, M.J.: Perceptual discrimination of computer generated and photographic faces. Digital Investigation **8**(3), 226–235 (2012). https://doi.org/https://doi.org/10.1016/j.diin.2011.06.003, https://www.sciencedirect.com/science/article/pii/S174228761100051X
6. Glasberg, R., Elazouzi, K., Sikora, T.: Video-genre-classification: recognizing cartoons in real-time using visual-descriptors and a multilayer-percetpron. In: The 7th International Conference on Advanced Communication Technology, 2005, ICACT 2005. vol. 2, pp. 1121–1124 (2005). https://doi.org/10.1109/ICACT.2005.246155
7. Harper, F.M., Konstan, J.A.: The movielens datasets: History and context. ACM Trans. Interact. Intell. Syst. **5**(4) (dec 2015). https://doi.org/10.1145/2827872, https://doi.org/10.1145/2827872
8. Ianeva, T., de Vries, A., Rohrig, H.: Detecting cartoons: a case study in automatic video-genre classification. In: 2003 International Conference on Multimedia and Expo. ICME '03. Proceedings (Cat. No.03TH8698). vol. 1, pp. I–449 (2003). https://doi.org/10.1109/ICME.2003.1220951
9. Ionescu, B., Lambert, P.: Classification of animated video genre using color and temporal information (2013)
10. Montagnuolo, M., Messina, A.: Automatic genre classification of tv programmes using gaussian mixture models and neural networks. In: 18th International Workshop on Database and Expert Systems Applications (DEXA 2007). pp. 99–103 (2007). https://doi.org/10.1109/DEXA.2007.92
11. Ng, T.T., Chang, S.F.: An online system for classifying computer graphics images from natural photographs. In: Delp, Edward J., I., Wong, P.W. (eds.) Security, Steganography, and Watermarking of Multimedia Contents VIII. Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series, vol. 6072, pp. 397–405 (Feb 2006). https://doi.org/10.1117/12.650162
12. Quan, W., Wang, K., Yan, D.M., Zhang, X.: Distinguishing between natural and computer-generated images using convolutional neural networks. IEEE Transactions on Information Forensics and Security **13**(11), 2772–2787 (2018). https://doi.org/10.1109/TIFS.2018.2834147
13. Roach, M., Mason, J., Pawlewski, M.: Motion-based classification of cartoons. In: Proceedings of 2001 International Symposium on Intelligent Multimedia, Video and Speech Processing. ISIMP 2001 (IEEE Cat. No.01EX489). pp. 146–149 (2001). https://doi.org/10.1109/ISIMP.2001.925353
14. Sankar, G., Zhao, H.V., Yang, Y.H.: Feature based classification of computer graphics and real images. 2009 IEEE International Conference on Acoustics, Speech and Signal Processing pp. 1513–1516 (2009)
15. Tan, M., Le, Q.: Efficientnet: Rethinking model scaling for convolutional neural networks (05 2019)
16. Zhang, R., Quan, W., Fan, L., HU, L.m., Yan, D.: Distinguishing computer-generated images from natural images using channel and pixel correlation. Journal of Computer Science and Technology **35** (02 2020). https://doi.org/10.1007/s11390-020-0216-9

Identifying and Analyzing Communities within a Social Network using Automatic Topic Labelling: Application to the Enron Dataset

M. Zakaria Kurdi

University of Lynchburg, VA

Abstract. A social network is typically made up of communities of users. Detecting such communities is an important task to identify users with shared interests and predicting their behavior. Several previous approaches to detect the communities based on the formal properties of the communities' subgraphs were proposed. The main assumption in this paper is that, within the context of a professional social network, communities are formed around a shared interest. Therefore, topics are used to identify the communities. This paper shows that the communities identified by topic do meet the formal requirements of a community by having a high modularity score. Besides, it is shown that creating communities based on topics improved our understanding of key aspects of the social network, like community overlap, User Topical Specialization (UTS), and topical seasonality.

Keywords Topic Modeling, Social Media Network Communities, Topic Seasonality, Community Overlap, User Topical Specialization

1 Introduction

Detecting the communities within a social network is an important task because it allows identifying users with shared interests and predicting their behavior. Depending on the type of the social network, there are several definitions of the concept of community. Formally speaking, a community within a social network, sometimes called a cluster, is a sub-network where there are dense links internally but that is connected with other communities with external links of lower density.

Different approaches have been proposed in the literature to identify communities or groups of users within social networks using formal criteria like the one above. Such approaches suffer from several limitations. First, they are not good at telling whether a user of the social network belongs to multiple communities (Xu and Hui, 2019). Second, these formal methods do not give insight about the nature of the interactions within a community or the

relationships between communities. Other approaches have used shallow definitions of topics, that are usually based on Latent Dirichlet Allocation (LDA) or one of its variants (see for example (Zhang et al., 2007) and (Yin et al., 2012)), therefore these approaches did not explore the full extent of the relations between topics and communities.

To overcome the above limitations, this paper explores using human interpretable topics as a means of community identification. Hence, communities are defined as a group of users that exchange information on a specific subject or topic. It is showed that topical communities also meet the formal requirements of a community, as this follows the intuition that people with the same professional specialty tend to exchange more information than people with different interests do. Even outside professional networks like the Enron email set that is investigated in this paper, it is hard to imagine the idea of community being entirely separated from the topics of interest of its users. In other words, regardless of the way a community is identified it is essential to understand the topics addressed within a community and their semantic relationships between each other. It is also essential to understand the relationship between the users and the topics. For example, at the sociocentric level, topics allow to easily measure and understand the overlap between communities as well as topic seasonality, in relation with the year calendar. At the egocentric level, using topics to detect communities allows measuring the degree of specialization of a given user and helps us understand the seasonality of the involvement of specific users in some tasks that are related to the used topics.

This paper is organized as follows. After presenting the existing literature in section two, section three presents the Enron Corpus and shows the steps followed to build the graph of the social network and automatically annotate it with topics. Sections four and five are about the analysis of the topical communities from a sociocentric and egocentric perspective, respectively.

2 Related Work

Several works focused on labelling social networks with topics. For example, McCallum et al. (2007) proposed the *Author-Recipient-Topic* model, a Bayesian network for social network analysis that discovers discussion topics conditioned on the sender-recipient relationships in the Enron corpus. This model combines the directionalized connectivity graph from social network analysis with the clustering of words to form topics using probabilistic language modeling. Cha and Cho (2012) discussed how to extend probabilistic topic models to analyze the relationships within the graph of social networks. They applied LDA and showed that it does not apply well to labelling popular nodes within the social network. Kalinowski and Kurdi (2019) compared the Term Frequency (TF) method, used as a baseline, with LDA combined with WordNet as well as two versions of conceptual

topic detection, both involving a version of keyword extraction combined with WordNet. The results showed that LDA combined with WordNet has the highest precision but a comparable F-measure to the conceptual approaches (around 60%). They also showed that topic labelling of social network shed light on both quantitative and qualitative relationships within the networks. Hagerer et al., (2021) showed that opinion-mining methods based on word embedding could be used to display correlated topic models on social media using an interactive visualization toolkit called SocialVisTUM.

Some works, like the one by (L'Huillier et al., 2011), used LDA to tag the texts of a forum with two categories. The tagged text is then used to identify and extract key-members within extremist groups. Others like (Sun and TY Ng, 2013) relied on a graph model of the most influential posts per topic. This graph is then transformed to a user graph to identify influential users.

Several algorithms have been proposed in the literature for community detection. One of them is based on betweenness centrality. This algorithm is an iterative approach to remove edges from the network to split it into communities. The removed edges are identified using *betweenness* measures (Newman and Girvan, 2004). Another method, called WalkTrap, it uses random walks on a graph to detect communities (Pons and Latapy, 2005). This algorithm starts with a selected node and goes to a neighboring node chosen randomly and uniformly. It then proceeds similarly to the next node. The number of steps of this algorithm is called the walk length. With an appropriate length, typically not too long, it can gather community information. Label propagation is another method, which is based on a semi-supervised machine learning that is initialized with a small subset of labelled data. The labels are then propagated iteratively to the unlabeled neighbors. The communities of edges will emerge under a common label (Raghavan et al., 2007). Matrix Blocking was used for community detection (Chen et al., 2012). This technique consists of constructing a tree hierarchy to order the nodes in a network and extract dense subgraphs as communities. Previous approaches to community detection suffer from a significant limitation, as they adopt strict borders between communities. This means that a user can only belong to a single community, which is not realistic in many social networks. To overcome this limitation, Saha et al. (2011) proposed a fuzzy clustering-based approach for community detection in a weighted graphical representation of social and biological networks. Another solution to this problem was proposed in (Xu and Hui, 2019). Their approach uses a method called the Partial Community Merger Algorithm (PCMA) with linear complexity. PCMA consists of three steps. First, it finds partial communities in the network, then it merges the related partial communities into completely merged communities, and finally it cleans up the noise accumulated in the merged communities to get the complete and cleaned-up communities.
Several works used a shallow sematic framework, typically LDA, to discover communities.

For example, Zhang et al. (2007) proposed a hierarchical Bayesian algorithm based on LDA to model communities as latent variables in the graphical model and are defined as distributions over the social actor space. Yin et al (2012) incorporated community discovery into topic analysis and proposed a community-based topic analysis framework called Latent Community Topic Analysis (LCTA). Their work assumes that community and topic are different concepts. Wang et al (2019) conducted a cross social network study to discover hot topics. First, they fuse text data from different social networks into one unified model. They get latent topic distribution from the unified model using the Labelled Bi-term LDA (LB-LDA). Based on the distributions, similar topics are clustered to form similar topics communities. Finally, based on the scores, the hot topics are chosen. In summary, no previous works have substantially explored the semantic nature of the communities or their relationships.

3 Network Building and Annotation with Topics

3.1 The Enron Corpus

Enron emails are a freely available corpus. The Federal Energy Regulatory Commission obtained it during its investigation of Enron's scandal[1]. This corpus is made of about 500.000 emails (1.32 GB of raw data) from 150 authors (Klimt et al., 2004), (Keila et al., 2005). This corpus is particularly relevant to this study because it represents interactions within a work environment, where it is easier to determine the key topics addressed during the exchanges between the employees.

3.2 Data annotation

A subset of 3800 emails from the Enron corpus were manually annotated (Kalinowski and Kurdi, 2020). A schema of seven topics was proposed: *meeting*, *management*, *IT*, *leisure*, *stock*, *accounting*, *legal*, and *other*, "*other*" being about emails of different topics that are not covered by the seven adopted topics. There are several unsupervised out off-the-shelf algorithms such as TF-IDF, K-means or LDA that can be used for topic annotation of texts (Kalinowski and Kurdi, 2019), (Halabi et al., 2010), (Blei et al., 2003). These approaches are generic by nature and do not give specific topics; therefore, they cannot provide the precise information needed for the network annotation. Hence, building a special module that is trained to annotate the texts with our set of topics is necessary. This is a challenging task because, as we can see in the figure 1, the adopted scheme does not cover all the emails: the category "other" has a large number of emails. Besides, some emails have content that spans over multiple topics.

[1] https://en.wikipedia.org/wiki/Enron_scandal

Given the significant imbalance of topic distribution, using machine learning to learn the topic becomes a challenge. To increase the accuracy, the topics *leisure* and *stock* were discarded because of the limited number of emails from these topics and because these topics are not central to understanding the work hierarchy within Enron.

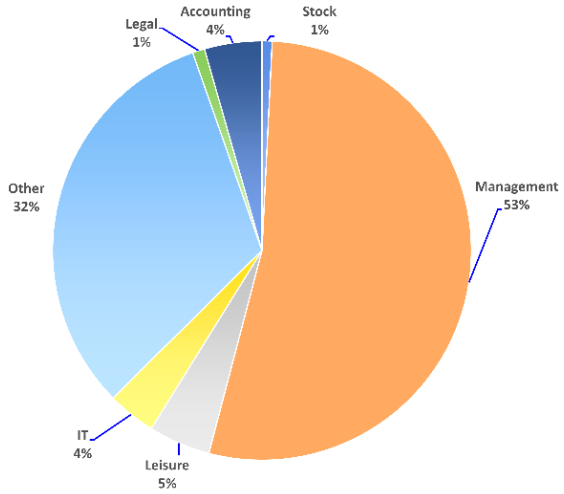


Fig. 1. Email distribution by topic in the Enron data set

3.3 Machine Learning Experiments

Several machine learning experiments were conducted. First, five classic machine-learning algorithms were used: Random Forest, Bagging, Bayesian Network, Logistic Regression, and Perceptron. The features used with these algorithms were keywords with the highest overall TF-IDF score. Experiments with 500, 1000, and 1500 keywords were conducted. The results of these experiments are presented in table 1.

Another experiment was conducted with a Convolutional Neural Network (CNN) with 300 embedding dimensions, 5 convolutional layers, and trained with 35 epochs. The f1 score was around 22%; the results were not high because of the limited size and the imbalance of the data.

The entire corpus was labelled with topics using the Random Forest Algorithm trained with 1000 words, which gave the best results. The network's graph was built after excluding all the emails with missing key information for the network, like date, sender, or destination. Therefore, the overall graph ended up being built with 405,018 emails. The topical graphs were built by considering the emails that were tagged with the targeted topic.

Table 1. F1 score of the classic machine learning experiment

# Key Words	ML Algorithm	F1
500	Random Forest	61.4
	Bagging	**62.2**
	Bayesian Net	31.4
	Logistic Regression	60.7
	Perceptron	28.9
1000	**Random Forest**	**65.8**
	Bagging	62.9
	Bayes	38.2
	Logistic Regression	61.6
	Perceptron	18.01
1500	**Random Forest**	**60.1**
	Bagging	58.9
	Bayes	38.0
	Logistic Regression	57.9
	Perceptron	26.0

4 Sociocentric Network Analysis of Topical Communities

Examining the sociocentric activities within the social network is a way to make general observations about the structural and functional tendencies within this network as a whole.

4.1 Quality of the Topical Communities

Using topics for network clustering into communities seems to be sound from an intuitive point of view. However, one might wonder if the topical clusters meet the formal requirement of being highly connected internally. To answer this question, modularity, a common criteria for evaluating the quality of clusters (equation 1) (Clauset et al., 2004), is used.

$$Q = \frac{1}{2m} \sum_{vw} \left[A_{vw} - \frac{k_v k_w}{2m} \right] \delta(c_v, c_w) \quad (1)$$

where A is the adjacency matrix of the network, m is the number of edges and $2m$ is the total number of stubs, k_v and k_w are the degrees of the nodes v and w respectively, $\frac{k_v k_w}{2m}$ is

the probability of an edge existing between the nodes *v* and *w;* the δ-function *δ(i, j)* is 1 if *i = j* and 0 otherwise.

The modularity of the entire network, broken down with betweenness and with topical communities is presented in table 2.

Table 2. Modularity of the communities

Topic/Community	***Q***
Overall (with betweenness)	0.586
meeting	0.731
accounting	0.888
management	0.612
IT	0.800
legal	0.689

According to (Clauset et al., 2004), a *Q* value greater than 0.30 is enough to claim there is a community. Hence, all the *Q* values in table 2 are significant enough to say that we have communities. The modularity of the topical communities is higher than the modularity of the overall graph. Also, specialized communities such as accounting and IT have a higher modularity than more generic topics like management.

4.2 Community Overlap and Inter-community Relationships

Two ways of measuring community overlap are proposed in this paper. First, overall overlap is measured as the network homogeneity. Homogeneity being about how evenly the emails are distributed over all the topics/communities. In a perfectly homogeneous network, the probability of an email belonging to any topic is the same. Hence, entropy is a natural measure of the overall homogeneity of the network (equation 2).

$$Hom(G) = -\sum_{t=1}^{|T|} p_t \log p_t \qquad (2)$$

where *G* is the graph of the network, |T| is the number of topics, and *t* is a given topic.

The homogeneity of the entire Enron network is 1.317. The theoretical maximum of the entropy is 1.609, which means that the Enron's network is highly homogeneous.

Another way to look at the community overlap is by examining the overlap between the pairs of communities. The interest in a topic *t* is measured as the number of emails sent by a given user μ about *t*. To measure the overlap, Goldberg and Kelley (2011) proposed using the cosine distance between key influential users within a community and a random user. This measure requires identifying the influential users, which is not an objective criterion,

since there are several alternative methods in the literature. In this paper, Kendall Tau rank distance is used. This distance is a metric that counts the number of pairwise disagreements between two ranking lists: τ1 and τ2. Therefore, the distance k(τ1, τ2) ranges between 1, when the two lists are identical, and -1, when they have reverse order (equation 3). Given its simplicity, this measure is more robust and provides a more objective account of the overlap than the cosine distance between influential users.

$$k(\tau_1, \tau_2) = \sum_{\{i,\ j\} \in p} \bar{k}_{i,j}(\tau_1, \tau_2) \tag{3}$$

where *P* is the set of unordered pairs of distinct elements in τ_1 and τ_2. $\bar{k}_{i,j}(\tau_1, \tau_2)$ is equal to zero when *i* and *j* are in the same order, and it is equal to one when they are in the opposite order.

To calculate the Kendall's Tau distance, the users are ranked within each community by their number of contributions. Then all the pairs of communities are compared. Since not all users are part of all the communities, the non-assigned users are given the lowest ranks. The results are presented in table 3.

Table 3. Kendall Tau distances between the topics

Community $_i$	**Community $_j$**	**Kendall's Tau**	**p Value**
accounting	IT	0.0329	3.30E-53
management	IT	0.0075	0.0004
legal	meeting	0.0075	0.0004
legal	IT	0.0071	0.0008
accounting	legal	0.0070	0.0010
accounting	management	0.0036	0.0869
IT	meeting	0.0034	0.1071
management	meeting	0.0024	0.2585
accounting	meeting	0.0021	0.3114
management	legal	0.0005	0.8148

Five pairs of communities have significant Kendall's Tau distances. Even in the case of significant distance, the value is not very big. This pattern is easy to explain given the relation between the topics. For example, the relationships between IT and accounting are the strongest, which is understandable given that accounting relies heavily on software. On the other hand, the people who post about IT don't necessarily post about meetings, that is why this pair is not significant.

4.3 Community Activity Seasonality

The goal of uncovering community activities seasonality is to show the underlying patterns behind every topic in relation to the calendar. The basic assumption here is that seasonality

is related to business activities typically achieved in specific months of the year. The graphs of the monthly averages of emails by topic are presented in figure 2.

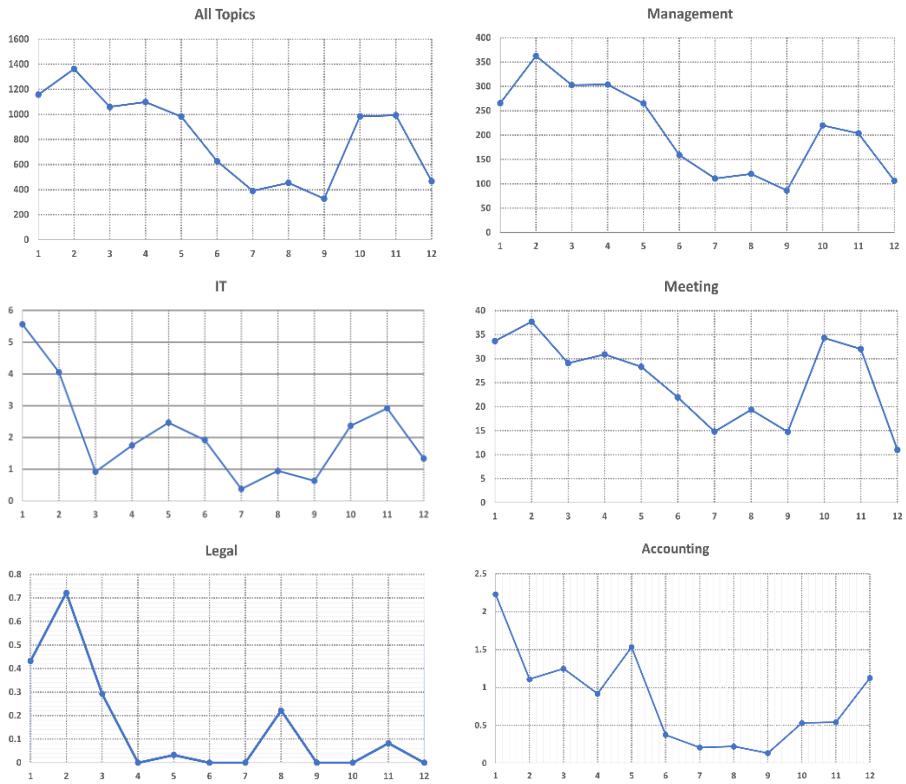


Fig. 2. Monthly averages of emails' numbers by topic

As we see in figure 2, although the seasonality of the emails by topic is affected by the general seasonality of the emails, we can also see that some topics have their own specificities. For example, in December the number of emails from all the topics decreases because of the holidays. A*ccounting* is the only exception in this month, which gets more emails because more work needs to be done at the end of the fiscal year. The same goes for January, where, unlike the general trend, accounting has higher activities because in this month accountants are typically very busy working on tax returns. *IT* is another exception for January where there is a significant increase of emails. This is related to the New Year's security instructions and updates. Besides, we are witnessing more regularity with the topics *management* and *meeting*, while the monthly differences are more extreme with the

topics legal, IT, and accounting. This shows that some topics have more substantial monthly seasonality than others do.

5 Egocentric Network Analysis within Topical Communities

Examining the social network at the level of a specific user helps identify those users who play an important role in a given process.

5.1 User's Topical Specialization (UTS)

Tagging the social network with topics leads to the key question of user topical specialization. It is essential to know who plays a pivotal role in a given process to understand the hierarchy within a social network. Let M = {m_1, m_2, …, m_n} be the group of messages sent by an active user μ. The user μ being active, we can assume that n>0. Let T = {t_1, t_2, …, t_m} be the topics addressed within the entire social network where m>=1. Let $\tau \subset T$ to be the subset of topics covered in the emails sent by μ. Hence, $|\tau|$ can have between 1 and *m* topics. Therefore, a user μ is said to be perfectly specialized if $|\tau|=1$, μ only sends messages about a single topic. The bigger the $|\tau|$ the less specialized is μ.

A measure of User Topical Specialization (UTS) is proposed in this paper (equation 4). In equation 4, if $|\tau|=1$, this is the highest level of topical specialization, so the entropy is zero and UTS=1 in this case. On the other hand, if $|\tau|=m$ then the entropy is maximal, which diminishes UTS considerably.

$$\text{UTS} = \frac{1}{\left(\sum_{i=1}^{|\tau|} -p(t_i) * \log\ p(t_i)\right) + 1} \tag{4}$$

The UTS values of some selected users are presented in table 4.

Table 4. UTS values of selected users and the number of emails per topic

email	UTS	Meeting	Acct[2]	Mgmt.	IT	legal
kay.mann@enron.com	0.756	331	15	3867	15	0
alan.comnes@enron.com	0.758	12	2	234	4	0
dana.davis@enron.com	0.720	3	0	20	0	0
infrastructure.ubsw@enron.com	0.577	19	2	44	0	0
brian.schwertner@enron.com	1	0	0	19	0	0
robert.lloyd@enron.com	1	0	0	160	0	0

[2] *Acct.* stands for accounting and *mgmt.* stands for management

As seen in table 4, some users are more focused on a limited number of topics while the emails of the others are scattered across the topics. The average UTS is 0.822 with a standard deviation of 0.196. This indicates that the users tend to be specialized but still most users contribute to more than one topic. An important conclusion that we can draw from table 4 is that belonging to a community is not necessarily binary, as the same user can belong to different communities with different levels of involvement.

5.2 Clustering Coefficient of a Node

The clustering coefficient (equation 5) is a measure of how the neighbors of a user's node within the social network's graph can form a complete subgraph or clique. Within the perspective of communities clustering, this coefficient allows one to see if a given user has a higher clustering coefficient within a given community than within the others. This is another way to compare the significance of the involvement of a specific user within every topic, except that here we do not get an overall measure like with UTS but rather individual scores for each community.

$$c_i = \frac{2L_i}{k_i(k_i - 1)} \tag{5}$$

where k_i is the degree of node i and L_i is the number of edges between the k_i neighbors of node i.

The clustering coefficients of five selected users are provided in table 5. As we can see in this table, most users have a higher clustering coefficient in some communities than in others. No user has a high clustering score within all the communities. For example, Julie Clyatt has a high clustering connection within *management* but not within the other topics. This confirms the tendency among the users of Enron's network to be specialized.

Table 5. Clustering coefficient of key users in the topic graphs

User's Email	Meeting	Accounting	Management	IT	Legal
kay.mann@enron.com	0.01	0	0.013	0	0
julie.clyatt@enron.com	0	0	0.044	0	0
jeff.dasovich@enron.com	0.013	0	0.013	0	0
john.arnold@enron.com	0.009	-1	0.017	0	0
phillip.allen@enron.com	0.011	0	0.011	0	0

5.3 User Topical Seasonality

Uncovering the interests of a user across a given period can give useful insights about the centers of a user's interests and activities. In figure 3, the percentages of involvement in the topics of six selected users are depicted during the twelve months of the year.

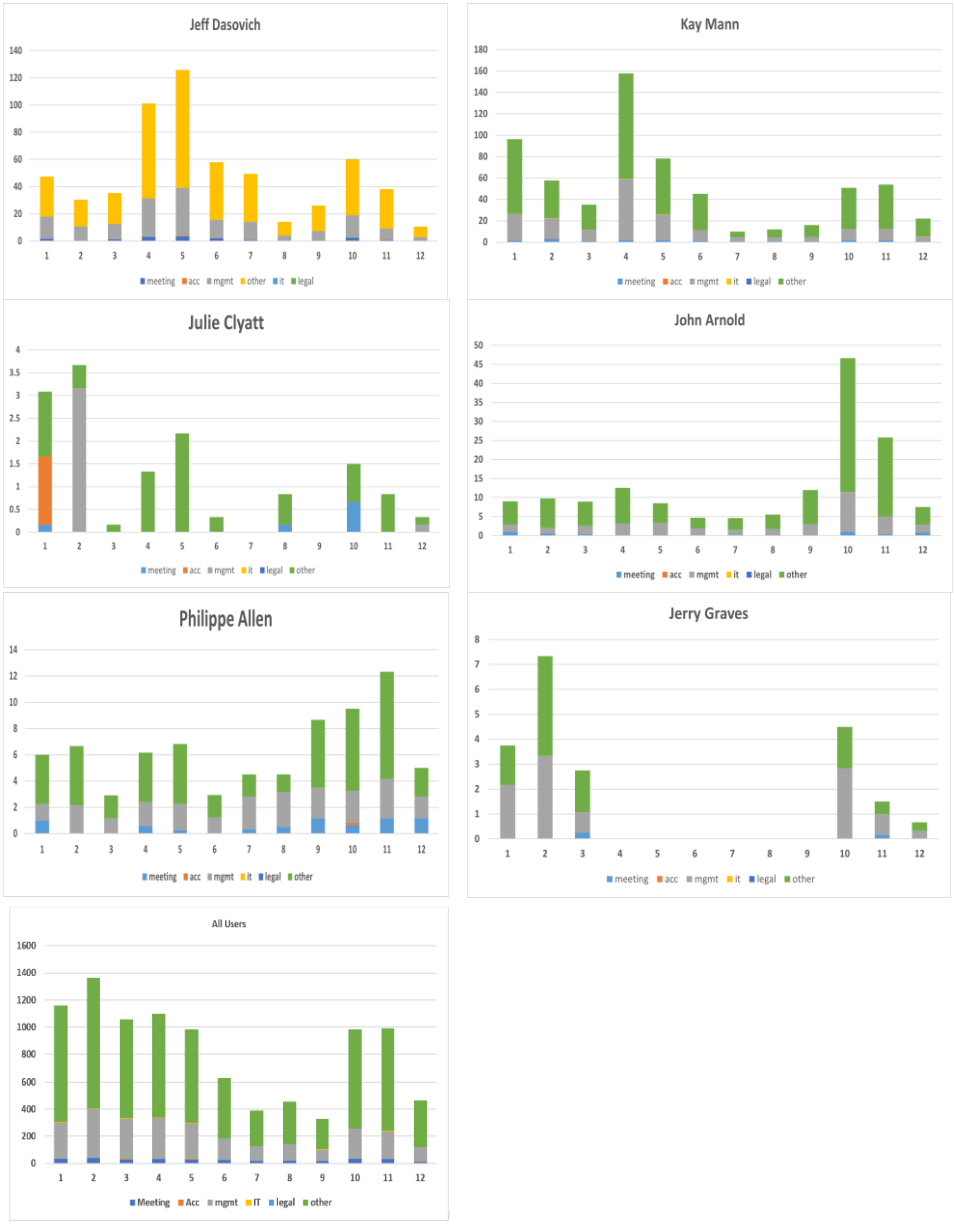


Fig. 3. Topical Monthly Seasonality of key Users

In figure 3, we can see that some users, like Jeff Dasovich have an almost constant interest in some topics across the year (management and meeting) while others have a seasonal interest in one topic. For example, Julie Clyatt is showing a diversified interest across the year, except that for accounting, she has a substantial involvement in accounting in January but not in any other month. Some users like Philippe Allen, show regular activity with some variation across the twelve months, while others like Jerry Graves are only active for six months. Although users are active over the twelve months, one can see a reduction in activity in the months of July, August, September, and December.

6 Conclusion

This paper is about using topics to identify communities within the Enron social network of emails. As shown in sections four and five, using topics to create the communities has several advantages compared to formal ones. First, it is naturally adapted to deal with overlapping users who belong to multiple communities. Second, given the known semantic relationships between the communities' topics, it is easier to understand the nature of the communities and their relationships with each other. At the sociocentric level, it was shown that the topical communities have a significant modularity score, which suggests that the topic community meets the formal connectivity requirement. Entropy and Kendall-Tau distance were used to provide insights about inter-community relationships. Finally, a measurement of the seasonality of the topics showed that the communities are not equally active across the months of the year and that there are some patterns that are easy to connect to the normal work calendar in the US. At the egocentric level, a new measure of user specialization was also proposed. This measure provides a quantitative evaluation of the dedication of a given user to a single community.

In the future, further investigation of other types of networks, such as Twitter, could help test the proposed approach on a different type of social network, where topics are broader. Topic communities can be refined using other NLP tools like sentiment analysis and opinion-mining. For example, within a network, one can identify a community that is interested in soccer. Then, within the soccer topic, supporters of specific teams can be grouped into sub-communities using opinion-mining techniques.

References

Blei, D., M., Ng, A., Jordan, M.: Latent Dirichlet Allocation. Journal of Machine Learning Research. Journal of Machine Learning Research 3 (4–5). 993–1022 (2003).

Cha, Y., Cho, J.: Social-Network Analysis Using Topic Models, SIGIR'12, August 12–16, Portland, Oregon, USA (2012).

Chen, J., Saad, Y.: Dense subgraph extraction with application to community detection. TKDE 24(7), 1216–1230 (2012).

Clauset A., Newman, ME, Moore C.: Finding community structure in very large networks. Phys Rev E Stat Nonlin Soft Matter Phys 70(6), 10-18 (2004).

Ding Z., Eren, M., Jia L., Lee G., Hongyuan Z.: Probabilistic Models for Discovering E-Communities, International World Wide Web Conference (IW3C2). Edinburgh, Scotland, May 23-26 (2006).

Goldberg, M. K., Kelley, S.: Overlapping Communities in Social Networks. Int. J. Soc. Comput. Cyber Phys. Syst. 1-2. 135-159 (2011).

Hagerer, G., Kirchhoff, M., Danner, H.: Pesch R., Mainak, G., Roy, A., Zhao, J., Groh G.: SocialVisTUM: An Interactive Visualization Toolkit for Correlated Neural Topic Models on Social Media Opinion Mining, arXiv:2110.10575, (2021) https://arxiv.org/pdf/2110.10575.pdf

Halabi, A. D., Islim, A., Kurdi, M. Z.: A hybrid approach for indexing and retrieval of archaeological textual information, In: International Conference on Knowledge-Based and Intelligent Information and Engineering Systems, pp 527-535 (2010).

Kalinowski, C., Kurdi, M. Z.: A Topic-based Forensic Analysis and Visualization of an Email network: Application to the Enron Dataset, Journal of Engineering, Science, and Computing (JESC) 1 (2), 1-22 (2019).

Keila P., Sillicorn, D.: Structure in the Enron email dataset. Journal of Computational and Mathematical Organization Theory, 1(11), 183–199 (2005).

Klimt, B., Yang, Y.: The Enron Corpus: A New Dataset for Email Classification Research, in Boulicaut J.-F. et al. (eds.), ECML 2004, pp. 217–226, LNAI 3201 (2004).

L'Huillier, G., Alvarez, H., Ríos, S. A., Aguilera, F.: Topic-Based Social Network Analysis for Virtual Communities of Interests in the Dark Web, ACM SIGKDD Explorations 12, pp 66-73. Washington, D. C. March (2011).

McCallum, A., Wang, A., Corrada-Emmanuel, X.: Topic and Role Discovery in Social Networks with Experiments on Enron and Academic Email. Journal of Artificial Intelligence Research 1(30), 249-272 (2007).

Pons, P., Latapy, M.: Computing communities in large networks using random walks. In: Yolum P, Güngör T, Gürgen F, Özturan C, (eds.). Computer and Information Sciences-ISCIS, pp. 284–293. Springer, Berlin (2005).

Raghavan, UN, Albert, R., Kumara S. Near linear time algorithm to detect community structures in large-scale networks. Physical Review E. 76(3), (2007).

Saha, T. Domeniconi, C., Rangwala, H.: Detection of Communities and Bridges in Weighted Networks. In: Proceedings of 7th International Conference on Machine Learning and Data Mining MLDM, New York, USA (2011).

Sun, B., TY Ng, V.T.: Identifying Influential Users by Their Postings in Social Networks. In: Atzmueller, M., Chin A., Helic, D., Hotho A. (eds) Ubiquitous Social Media Analysis. MUSE. Lecture Notes in Computer Science, vol 8329. Springer, Berlin, Heidelberg. (2013)

Wang, X., Zhang, B., Chang, F.: Hot Topic Community Discovery on Cross Social Networks, MDPI Future Internet, 11(3), pp 60-76 (2019).

Xu, E. H., Hui, P.M.: Uncovering complex overlapping pattern of communities in large-scale social networks. Appl Netw Sci 4, 27 (2019).

Yin, Z., Cao, L., Gu, Q., Han, J.: Latent Community Topic Analysis: Integration of Community Discovery with topic Model, ACM Transactions on Intelligent Systems and Technology 3(4) September Article No.: 63 pp 1–21 (2012).

Zhang, H., Qiu, B., Giles, C. L., Foley, H. C., Yen, J.: An LDA-based Community Structure Discovery Approach for Large-Scale Social Networks, IEEE International Conference on Intelligence and Security Informatics, 23-24 May, New Brunswick, NJ, USA (2007).

Maize Yield Predictive Models and Mobile-based Decision Support System, for Smallholder Farmers in Africa

Chollette Olisah[0000-0001-7909-0384], Lyndon N. Smith and Melvyn L. Smith

Centre for Machine Vision, Bristol Robotics Laboratory, Department of Engineering Design and Mathematics, University of the West of England, Bristol, BS16 1QY, UK
chollette.olisah@uwe.ac.uk

Abstract. Existing machine learning models for crop yield prediction, model environmental data on the assumption that soil variables are unaffected by weather variables, and therefore learn their intrinsic features independently. If the focus of crop yield prediction is aimed at supporting smallholder farmers in making farming decisions, then modelling the environmental variables independently might not be informative for the farmer. In this paper, we propose a comprehensive machine learning based crop yield decision support tool for smallholder farmers. It comprises of predictive machine learning models that model the dynamic interactions between environmental variables, for predicting crop yield at the level informative to a smallholder farmer. Then, the best model is integrated to a mobile application with farmer education and market access modules, to provide the smallholder farmer with a tool that enables him/her to 'farm smart'. From evaluation of our random forest regressor (RFR), extreme gradient boosting regressor (XGBoostR), and multi-layered perceptron regressors (MLPR), using the mean squared error (MSE) metric which quantifies the average of the square of the error, the values of 0.0075 t/ha, 0.1416, 0.3031 t/ha were achieved, respectively. This shows that the RFR model provides the minimal error between the predicted and ground truth crop yield values.

Keywords: maize, crop yield prediction, machine learning, decision support system, Sub-Saharan Africa

1 Introduction

Although over 60% of the population of Sub-Saharan Africa are engaged in small scale farming [1], agriculture only contributes about 23% of the region's GDP [1]. The inability of Africa's smallholder farmers to substantially drive Africa towards food security as expected [2], might be a consequence of socio-economic and political factors at the secondary level [3]. Additional factors may include a lack of farming education, limited access to market, lack of precision driven farming technology, and others at the primary level [4]. In this paper we argue that the smallholder farmer can leverage technology available to him/her to overcome farming limitations arising from factors at the

primary level. Subsequently, we discuss the technological advancement in precision farming and examine the current role of mobile technology in agriculture as it pertains to the smallholder farmer. The paper is structured as follows: Section 1 outlines the relevant background situation in African farming, in section 2 we describe the methodology employed, while section 3 covers the modelling undertaken and results obtained. Finally, in section 4 the Conclusion, we summarize the findings of the paper.

2 Background and way forward

Technological developments are currently enabling radical transformations in precision agriculture. Technologies in the form of robots [5], sensors [6], drones [7], and data analytics through machine learning [8-16] are driving agriculture towards smart farming. Most of these technologies are costly and out of the reach of smallholder farmers. However, data analytics through machine learning can provide the smallholder farmer with inexpensive decision systems that can help him/her to estimate yield during a crop growing season. There are two main directions in crop yield predictions: plant genotype data driven yield prediction [8, 10] and environmental data driven yield prediction [11-16], or a combination [9]. The genotype data is concerned with the interaction of plant genes and the environment and the results for farm produce. Genotype data are scarce, and also require rigorous collection and analysis, which may present a challenge for many farmers. On the other hand, environmental data can be made readily available to farmers, and offers rich information useful for crop yield prediction.

2.1 Machine learning for modelling crop yield

Machine learning algorithms have in recent years been increasingly applied to uncovering the nonlinear relationship of environmental variables as predictors of crop yield. In 2016 Jeong et al. [11] trained the random forest (RF) machine learning algorithm for crop yield prediction for evaluating food security at the regional and global scales. Later, in 2020, Alhnaity et al. [12] applied the Long Short-Term Memory (LSTM) for predicting tomato yield for crops grown indoors in controlled environments. In the same year Khaki and Wang [13] trained a deep neural network (DNN) for predicting maize yield. Their DNN comprised of 21 hidden layers with 50 neurons in each layer. Training employed data of 142,952 samples of maize plant genotypes, along with weather and soil variables. In [14] two convolutional neural networks (weather convolutional neural network W-CNN) and soil convolutional neural network S-CNN) were created for modelling temporal and spatial data for weather and soil. The extracted features for weather and soil, emerged from the fully connected (FC) layer of the network. Then, historic crop yield, modelled as a time dependent variable, along with the output of the FC layer and management data, were fed to a recurrent neural network (RNN) for forecasting yield. In [15] a feature selection approach is used to identify important weather, soil, and management variables, and their interactions, which were fed to a multiple linear regression model for predicting corn and soybean yield. An ensemble machine learning model that combines linear regression, LASSO regression, Extreme

Gradient Boosting (XGBoost), LightGBM, and random forest was proposed in [16] for corn yield prediction. In 2021 Shahhosseini et al. [17] modelled yield by building two convolutional neural networks, W-CNN and S-CNN for modeling weather and soil respectively, and a deep neural network (DNN) for representing the management data. The latter were collected at the fully connected (FC) layer of the network. This model was used to construct a homogeneous ensemble model by varying the data set using bootstrap sampling while the heterogenous ensemble model was created with the use of different hyperparameters.

Most of the literature has modelled crop yield under the assumption that environmental variables occur independently of each other. For instance, they assume that soil is unaffected by weather and therefore are focused on learning the intrinsic features of weather and soil variables independently. However, the interaction between climate and soil is an integral part of plant growth [18]. Also, previous work has approached crop yield from a regional or global perspective, with the goal of informing food policies and providing a mechanism for countries or The World Bank to use in appraising food security. However, if smallholder farmers are provided with predictions of crop yield, this can empower them to make better farming decisions, such as determining whether a given farmland has the potential to meet the anticipated yield capacity for a given planting season. Therefore, in this study we propose a comprehensive machine learning based crop yield decision support tool for smallholder farmers. It comprises of 1) a crop yield predictive model developed with machine learning techniques and combining environmental data to model the dynamic interactions between environmental variables, and 2) a mobile application for farmer education and market access which integrates predictive models to provide the smallholder farmer with a tool that enables him/her to farm smart at a low cost. The benefit that the proposed system provides for the farmer is the ability to utilize information from his/her farm for determining the best ways to maximize crop yield, thereby eventually improving and maintaining food production. Most farmers in Africa currently lack the ability to reach the market directly without third party buyers. The proposed system's market access platform will help smallholder farmers manage farm produce from farm to the market, in order to prevent food waste.

3 Methodology

3.1 Study region

We used Nigeria as the case study for Africa. Nigeria [9.0820° N, 8.6753° E] is located on the west coast of Africa with an arable land of 341 million [19]. Around 87 percent of Nigeria's households in the rural regions are smallholder farmers [20] who farm at least one of the major crops in Nigeria, (the latter comprising maize, cassava, guinea corn, yam beans, millet, and rice, and half of these farmers are corn growers). There are 36 states in Nigeria, of which the most and least number of districts in each state are

214 and 10, respectively. The data considered in this study are weather, soil, maize crop yield, and hectare. Further details relating to the data include:

- To extract the environmental data from the grided soil and climate Africa maps, each state's latitude and longitude information were retrieved from Google Maps.
- Historical climate data were collected from WorldClim [21], which is a database of high spatial resolution global weather and climate data. Their grided map data is comprised of eight weather variables, namely: average temperature, minimum temperature, maximum temperature, precipitation, solar radiation, wind speed, and water vapor at 30s, 2.5m, 5m and 10m spatial resolutions from ~1 km^2 to ~340 km^2. At each grid point, monthly weather data (that span a 30 year period between January 1 to December 12 of each year) are given as averaged readings per resolution.
- We collected historic yearly corn yield data, along with the data on the size of the cultivation area of land measured in hectares, from 1960 to 2006 for each of the 36 states of Nigeria from [22], under a subscription.
- The grided soil data provided by AFSIS [23] at spatial resolution of up to 250m included wet soil bulk density, dry bulk density, clay percentage plant available water content, hydraulic conductivity, upper limit of plant available water content, lower limit of, organic matter percentage, pH, sand percentage, and saturated volumetric water content variables measured at depths 0–5, 5–10, 10–15, 15–30, 30–45, 45–60, 60–80, 80–100, and 100–120 cm. Using the longitude and latitude information, the point soil values at specific district locations for each of the 36 states in Nigeria were extracted. This data spanned between 1960 to 2012.
- We used ARCGIS professional 2.5 to extract the point values for the climate and soil environmental data from each district within the 36 states of Nigeria.

3.2 Data Preprocessing

Given that data were acquired from different places, there were inconsistencies in the number of years between the climate, soil, hectare and crop yield data. For instance, the historic crop yield data covered between 1995 to 2006 and soil between 1960 to 2012. There was also the problem of missing values and numerous variables, of which only a few might be relevant to the task at hand. For these reasons we applied statistical and machine learning techniques to address these problems. Below are the algorithmic steps used in preparing the data.

The Algorithmic Steps

i. Average the values of weather, and soil variables across spatial resolution and the measured depth, respectively.
ii. Forecast crop yield and hectare by six timesteps to correspond with the last timestep of soil data using the autoregressive integrated moving average (ARIMA). Then, average into a single value for a state.

iii. Merge all the environmental data into a single dataset, remove missing values and manually select variables that can easily be measured by the smallholder farmer. After missing values were removed, only 23 states which are (Abia, Abuja, AkwaIbom Anambra, Bayelsa, Benue, Crossriver, Delta, Ebonyi, Edo, Ekiti, Enugu, Imo, Kebbi, Kwara, Lagos, Ogun, Ondo, Osun, Oyo, Plateau, Rivers, Taraba) environmental variables are represented.

iv. Measure the association between environmental variables from (iii) to identify variables highly associated with crop yield using the Kendall correlation (KC) technique.

3.3 Predicting Future Crop yield

We used the ARIMA model for forecasting crop yield in six (6) year timesteps. The ARIMA model is a simple machine learning technique that lends itself to solving non-complex time series prediction problems. By non-complex, it is meant that the time series data is non-seasonal and exhibits patterns. This model combines both auto regressive (AR) and moving average (MA) models to predict future points for a given time series based on its past observations and random errors. ARIMA is characterized by three terms: p is the AR order term which signifies the number of prior values to be used as predictors in the model, d is a parameter for making the series stationary, and q is the MA order term which indicates the number of forecast errors of prior values the ARIMA Model needs in order to predict future points in the series. The combination of the AR and MA models to form a ARIMA model is mathematically expressed as [24]:

$$\hat{Y}_t = \alpha + \beta_1 Y_{t-1} + \beta_2 Y_{t-2} + \cdots \beta_p Y_{t-p}\, \epsilon_t + \emptyset_1 \epsilon_{t-1} + \emptyset_2 \epsilon_{t-2} + \cdots + \emptyset_q \epsilon_{t-q}$$

where α is a constant, ϵ_t is the error term at time *t*, $\beta_1, \beta_2, \beta_3, \cdots, \beta_p$ are the AR parameters, and $\emptyset_1, \emptyset_2, \emptyset_3, \cdots, \emptyset_q$ are the MA parameters and are combined to form $\hat{Y}_t$, the future points in the series.

Typically, to predict future data points of a time series data using the ARIMA model, the stationarity of the series data will have to be tested using the Augmented Dickey Fuller (ADF) test [25]. The null hypothesis of the ADF test indicates a non-stationary series. However, if the p-value of the test is less than the significance level (0.05) then the series is considered stationary. With a given stationary series, the values of p and q terms can be determined by observing the values above the significance limit of the partial autocorrelation function, and autocorrelation function plots [25], respectively.

The crop yield and hectare values covered 12 years for the 36 states of Nigeria. The (p,d,q) terms were modified based on the ADF test, partial autocorrelation function plot, and autocorrelation function plot for each of the state series values. For instance, the (p,d,q) terms for predicting crop yield and hectare future data points for Ogun, Anambra, and Taraba states are (5,0,1; 5,0,1), (2,1,1; 1,0,1), (2,1,1;10,1) given their p-

values of (0.000000; 0.000000), (0.343881; 0.997572), and (0.450462, 0.859081) respectively. The p-values show that only Ogun state's series values were stationary, whereas Anambra and Taraba states values were non-stationary and required a differencing to make the series stationary. The hectare series were normalized by logarithmic transformation due to high level of skewness before applying ARIMA and were reverted afterwards.

Feature selection
Feature selection is a crucial step necessary for helping a machine learning algorithm to learn the intrinsic features of a dataset, prevent the model from overfitting to noisy features in the training data, and to generalize well with unseen test data. We initially selected some environmental variables that are easily accessible to the smallholder farmer based on the literature [26-28]. Then, we further statistically quantified the relationship that each of the selected variables have with crop yield by applying the KC [29] technique. The KC is a non-parametric correlation technique which measures the strength of association between two variables. It is useful to this study's task of feature selection where the interest is to quantify the associative relationship between crop yield and each of the weather variables, soil variables, and hectare. KC is used because the dataset is highly skewed due to outliers from crop yield and hectare.

Designing a Predictive Model using Machine Learning Techniques for Tabular Data
We employed machine learning algorithms well known within the machine learning community. They are the XGBoost, RF, and DNN. The predictive power of XGBoost, RF, and DNN were explored for crop yield prediction, though within a framework for forecasting crop yield at various timesteps. However, our objective is not to forecast crop yield for a given time frame but to predict the outcome of a farm produce given some known environmental data. Therefore, given that a ML algorithm's performance is subject to the data to be learned, the RF and XGBoost ML algorithms were tuned and the architecture of the DNN designed in a way that scaled to the regression problem. Of the 1380 sample points of the dataset, 1281 are used for training the models, with 275 and 274 for validating and testing the models, respectively.

Extreme Gradient Boosting Regressor
XGBoost [30] is a supervised learning algorithm that has been widely applied to solving regression problems. It is an extension of the gradient boosting decision tree (GBDT) algorithm [31] which learns a task by building multiple decision tree models organized in sequences, with the goal of minimizing the prediction errors of new models in the sequence based on the errors of previous models. Hence, a strong model of decision trees is formed. The XGBoost uses advanced regularization, L1 norm and L2 norm, to speed up the GBDT model generalization process.

The hyperparameters of the XGBoost regressor in this study were selected based on the grid search hyperparameter tuning method. The number of estimators is set at 1000, which is the number of trees the XGBoost algorithm built from the training set. Maximum depth was assigned a value of 10; this controls how specialized each tree is to the

training dataset. The higher the value the more likely the XGBoost algorithm overfits to the training set. Learning rate had a value of 0.1, while the minimum samples split was assigned a value of 2. This is a parameter that specifies the minimum number of samples required at a leaf node to enable splitting to occur. Learning rate, which controls the speed at which new trees can make corrections to the error of previous trees, was assigned to a value of 0.1. Subsample was given a value of 1. This parameter signifies the number of training set samples XGBoost uses to grow the trees.

Random Forest Regressor
Random forest regressor (RFR) is a supervised learning algorithm that relies on a random ensemble of several decision trees [32] to make a regression decision. Each tree is built based on bagging and feature randomness to ensure that it is focused on unique aspects of the data while maintaining low variance without increasing the bias of the decision tree. RF can handle thousands of variables of different types.

In this study, using the grid search method for hyperparameter tuning, the following parameters were selected as the best performing hyperparameters: Number of estimators is set to 100: The number of trees the RF builds. Maximum depth assigned a value of 10: Controls how specialized each tree is to the training dataset. The more the value the more likely the RF algorithm overfits to the training set. Minimum samples split assigned a value of 2: is a parameter that specifies the minimum number of samples required at a leaf node to enable splitting to occur. Minimum samples leaf with a value of 1: this is the minimum acceptable number of samples required to form a leaf node.

Multilayered Perceptron Regressor
The multilayered perceptron (MLP) is a class of networks that belong to the family of artificial neural networks [33]. Given that most real-world structured data problems do not follow a known distribution, MLPs have been well suited to solving them since they approximate functions that are separable non-linearly. Typically, an MLP consists of three layers – the input, hidden and output layers. For any neuron, except neurons at the input, each layer passes data to the next layer. The output of a node must be above a specified threshold value defined by an activation function. We formulate the neuron activation function as:

$$f(x) = \varphi\left(\sum_{i}^{n} w_i x_i + b\right)$$

where x is the input, w_i is a weight associated to a neuron, b is the bias associated with a neuron, and φ is the activation function. During learning, the network updates its weight, w_i and bias, b using the backpropagation method to minimize the difference between a target output of a problem and the network's predicted output.

There is no one-fit-all MLP architecture because different problems demand different architectural settings or network hyperparameter tuning. Therefore, we designed a 4-layer fully connected MLP architecture which comprises an input layer with neurons the length of columns in the tabular data, two hidden layers of 16 neurons each, and an output layer that predicts a numeric value that represents crop yield. The training set was scaled into a distribution centered around zero-mean and one-standard deviation and is used to transform every input into the network. We used the Glorot uniform (GU) [36] weight initializer to define weights applied to the input and hidden layer's neurons along with zero bias. A rectified linear unit (ReLU) activation function [37] is used for deciding whether to pass an input/hidden layer's neuron or not to pass it. The ReLU is commonly used because it can overcome the vanishing gradient problem. The hyperparameters of the proposed MLP architecture are described: The learning rate is set to 0.001 using the Adam optimizer which controls how the network updates weights with respect to the loss gradient. Epoch is set to 100 to enable the network to make 100 passes through the entire training set during which the neuron weight is updated 41 times based on 32 samples in batch size per update.

3.4 Evaluation

The predictive models are evaluated with 274 data sets of the test set, using the mean absolute error (MAE), mean absolute percentage error (MAPE) and the mean squared error (MSE), which are described as follows.

Mean Absolute Error (MAE): is a statistical metric for evaluating the average magnitude of the errors between observations. This metric is commonly used for evaluating a regression model's prediction performance and it is useful for our regression problem because it is known to be robust to outliers; particularly of types that are common in crop yield data while being necessary for yield prediction. Mean Squared Error (MSE): is another commonly used statistical metric for evaluating regression models. The prediction error is again the difference between the true value and the predicted value, but it differs from other metrics in that the mean differences between observations are squared. Mean absolute percentage error (MAPE): The MAPE is also known as the mean absolute percentage deviation error. It is a measure of accuracy of a regression model that is commonly employed in machine learning because it offers a convenient and straightforward interpretation of the degree of error between instances of the test set.

4 Results and Discussions

The loss in the performance of the MLPR model fitting of the training data and a fit to new data (via validation set) as it updates weights through a backpropagation algorithm is shown in Fig. 1. Here it can be observed that the MLPR model is able to generalize well on the validation weather and soil data. Thus, it is expected that the model will perform well on test data sets that were unseen during training.

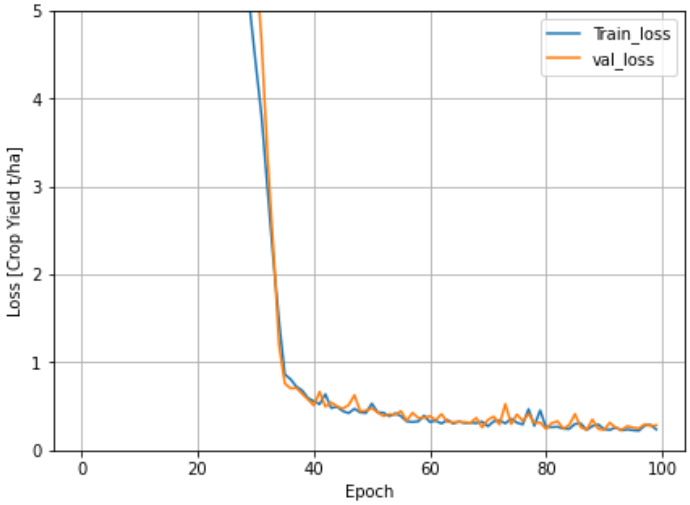


Fig. 1. Trend graph for training and validation loss of the MLPR, for modelling of crop yields in Nigerian farms.

Fig. 1 shows that the smaller the prediction errors, the closer the predicted sample points are to the line of best fit. This means that the RFR, XGBoostR and MLPR models can be employed for crop yield prediction given weather and soil environmental variables, though the RFR model performed best. The MSE indicates that the RFR and MLPR with values 0.0075 t/ha and 0.1416 t/ha, respectively, were both better than the XGBoostR model in minimizing the prediction errors (both across a dataset and the difference in error between a predicted crop yield value and its ground truth value). Even though XGBoostr achieved a low prediction error of 0.3031 t/ha, it is higher than the error of RF and MLP models, possibly because the model's variance is higher. However, when using the MAE to reflect the average bias of a prediction model as a measure of performance, the XGBoostR prediction error is observed to be lower. In general, it can be stated that the MAE of RFR and MLPR models show higher bias over the variance, while the XGBoostR model shows higher variance over bias. Further interpretation of the MAE of the models on the test set samples with a total of 275 envi-

ronmental data points, reveal that about 2%, 14%, and 71% of the data samples comprise the deviation (no matter how small) between the predicted value and the ground truth values for the RFR, XGBoostR and MLPR models, respectively.

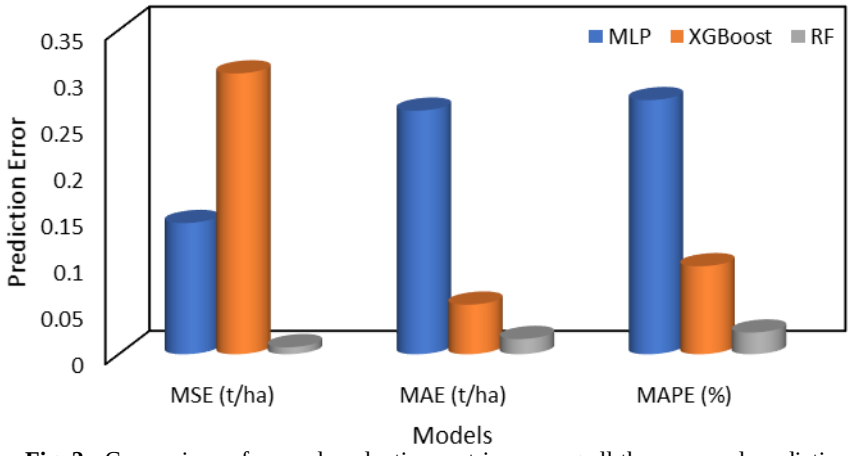


Fig. 2. Comparison of several evaluation metrics among all the proposed predictive models for crop yield prediction for Nigerian farms.

A more intuitive way to observe the models' performance, is through MAPE which quantifies the percentage of the prediction error. The RFR, XGBoostR, and MLPR models gave average prediction errors of +/-0.024%, +/-0.095%, and +/-0.2743%, respectively. Overall, the RFR model maintained significantly lower errors between the predicted crop yield (tonnes per hectare) values and ground truth values, than the XGBoostR and MLPR models. The performance of the models observed in this study is likely to be associated with the gradual changes, between values of grided data points, of weather and soil variables for across the 23 states of Nigeria.

Further analysis included a unit testing of two sample points from the unseen test set to investigate deeper into the performance of the model. From the test set a sample point of environmental values were retrieved for a farm in Abia state, Nigeria: 26.24211, 193.5, 1.6860886, 5.25, 28.166666, 63.166668, 10.333333, 41.39817, which comprise: temperature, precipitation, windspeed, soil pH, clay, sand, silt, and hectare, respectively. The models predicted 58.32268137 t/ha, 58.324867 t/ha, and 58.94982 t/ha, which is a difference of -0.00000137 t/ha, -0.002187 t/ha, -0.62714 t/ha for the RFR, XGBoostR, and MLPR models, respectively. Using environmental variable data points assumed for a farm in Osun state, Nigeria: 26, 18, 2, 3.4, 25.83, 66, 13, 0.5778 representing temperature, precipitation, windspeed, soil pH, clay, sand, silt, and hectare; the RFR, XGBoostR, and MLPR models predicted 12.48870942 t/ha, 9.6671715 t/ha, and 61.549477 t/ha, respectively. This result show that the error of the MLPR can be unexpectedly very high which might limit its applicability in real-world use.

4.1 Mobile Application of Predictive Modelling for Smart Farming

In Africa, there is currently minimal employment of drones, GPS systems, robots and other advanced technologies, in agriculture. However, there is a growing wide coverage of mobile phone usage which has been identified as a technological tool for enhancing the experience of farmers in terms of education, and access to the market, as identified in [38]. In this paper, we propose to utilize mobile phones as a practical tool for improving the farming experience on the ground in Africa. We designed and developed a mobile application, termed "IntelliFarm", which includes key functionalities: 1) a farmer-market communication interface to enable the farmer to directly access buyers in a timely way before farm produce decomposes, and 2) an education interface to house information on best farming practices that can be useful to a smallholder farmer, and 3) the predictive model interface for determining, ahead of a planting season, the produce to be expected from a given farm land. This was presented to the farmers in terminologies the smallholder farmer will understand best at a rural area, which is "number of bags of corn" per hectare. How the farmer and buyer interact in the developed mobile application can be visualized in Fig. 3. The *IntelliFarm* application will serve as a supporting tool for aiding farmers' decisions. Considering that the soil variables contribute substantially towards the model's prediction, if a farmer's prediction is lower than expectation, he/she can search through the education module for possible ways to balance the soil pH by adding fertilizers, or by investigating the nature of the soil, as ways of improving on crop yield.

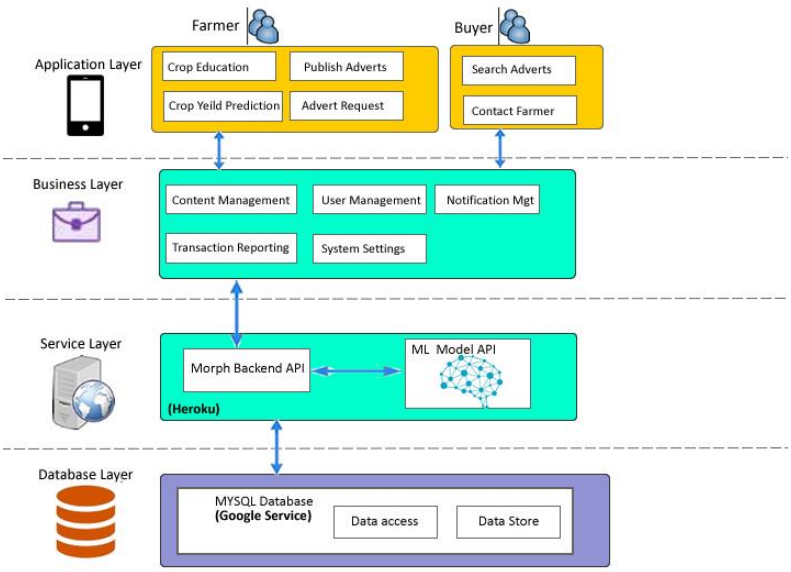


Fig. 3. The IntelliFarm mobile application architecture. Each stage represents: the application layer, business layer, service layer and the last is the database layer.

Fig 4 shows IntelliFarm's user graphical interfaces for: farmer registration and login, welcome and core modules of the application, the prediction module, and education and market access.

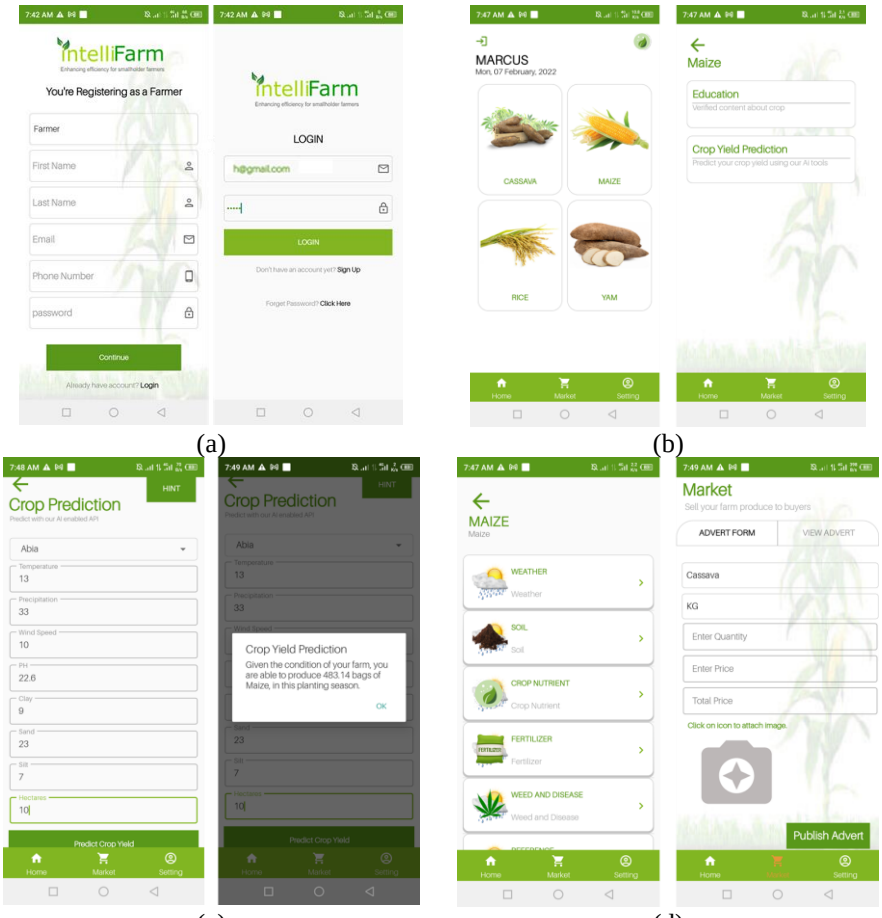


Fig. 4. The graphical user interfaces of IntelliFarm core functionalities. (a) farmer registration and login interface, (b) interfaces for welcome, and core modules of the application, (c) the prediction module interface, and (d) the education and market access interfaces of the application.

As shown in Fig. 3, the mobile application is comprised of a number of layers. At the application layer, there are two main users of the system, the farmer and the buyer, who can sign up and login to access the application and perform core processes. The business layer is the backend of the application, and it is concerned with ensuring that the user accounts, notifications, and application usage reports, are generated and stored. At

the service layer, Morph's backend is currently hosted under a free license on the Heroku cloud, where the predictive model is stored. The model takes temperature, precipitation, windspeed, soil pH, soil clay percentage, soil sand percentage, soil silt percentage, and size of farming area in hectare as input to be provided by the farmer from real-time measurement on the farm. The weather data (temperature, precipitation and windspeed) can be provided through a third-party weather application programmer interface (API), though it was not considered in this paper. The soil data can be measured by the farmer using techniques which can be easily configured by a smallholder farmer using local materials. An example of a local material and the process of achieving the measurement is currently provided in the education tool. Once the farmer can retrieve the required values, the farmer can proceed to click the prediction button, which will push a prediction request to the third-party machine learning model API to process and return crop yield predictions back to the mobile application in a format useful to the farmer. Finally, the database layer is a database management system, mySQL, which enables the user to access and store information currently stored in the cloud hosted by Google Services. The application user manual is provided in the application to help smallholder farmers navigate the application with ease.

5 Conclusion

In this study we designed models for predicting crop yield in a way useful to the smallholder farmer during the planting season. RFR and XGBoostR models were designed using grid search hyperparameter optimization, to generate hyperparameters and their values, to enable the model to fit the data more accurately. A multi-layered perceptron regressor (MLPR) was designed by experimenting with various depths of hidden layers and hyperparameters. In model evaluation using test data unseen during training, the RFR, XGBoostR and MLPR models achieved MSE values of 0.0075 t/ha, 0.1416, 0.3031 t/ha, respectively. Based on this performance, the RFR model was integrated into the developed mobile application, along with functionalities for farmer education and market access, thereby enabling a smallholder farmer to farm smart. It is expected that widespread adoption of this technology in Nigeria, and across Africa more generally, could assist with attaining significant increases in crop production, which would greatly benefit the region both in terms of food security and economically. In the future work, the predictive model will be extended to handle predictions for other staple food such as cassava, yam, and rice. It will also be worth including a third-party API to generate weather data automatically for the farmer. Subsequent studies will include usability analysis of the application after initial roll-out to quantify "the smallholder farmers perceived ease of use" of the application.

Acknowledgement

We gratefully acknowledge support from the UK GCRF/BBSRC Agri-tech Catalyst Seeding Award 'Artificial Intelligence (AI) Enhanced Smallholder Farm Software

Tool' (grant number 87645); and thank Morph Innovations Limited for the seamless industrial partnership during the development of the mobile application. We also acknowledge Baze University Nigeria, for providing an enabling environment for collecting and studying the environmental variables used for the project.

References

1. Lutz Goedde, Amandla Ooko-Ombaka, and Gillian Pais. Private-sector companies can find practical solutions to enter and grow in Africa's agricultural market. https://www.mckinsey.com/industries/agriculture/our-insights/winning-in-africas-agricultural-market [Accessed 29 01 2022]
2. Fan, S. and Rue, C., 2020. The role of smallholder farms in a changing world. In The Role of Smallholder Farms in Food and Nutrition Security (pp. 13-28). Springer, Cham.
3. Stringer, L.C., Twyman, C. and Gibbs, L.M., 2008. Learning from the South: common challenges and solutions for small-scale farming. Geographical journal, 174(3), pp.235-250.
4. Amare, M., Cissé, J.D., Jensen, N.D. and Shiferaw, B., 2017. The Impact of Agricultural Productivity on Welfare Growth of Farm Households in Nigeria: A Panel Data Analysis. FAO. Rome.
5. Rose, D.C., Lyon, J., de Boon, A., Hanheide, M. and Pearson, S., 2021. Responsible development of autonomous robotics in agriculture. Nature Food, 2(5), pp.306-309.
6. Kulbacki, M., Segen, J., Knieć, W., Klempous, R., Kluwak, K., Nikodem, J., Kulbacka, J. and Serester, A., 2018, June. Survey of drones for agriculture automation from planting to harvest. In 2018 IEEE 22nd International Conference on Intelligent Engineering Systems (INES) (pp. 000353-000358). IEEE.
7. Sushanth, G. and Sujatha, S., 2018, March. IOT based smart agriculture system. In 2018 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET) (pp. 1-4). IEEE.
8. Shook, J., Gangopadhyay, T., Wu, L., Ganapathysubramanian, B., Sarkar, S. and Singh, A.K., 2021. Crop yield prediction integrating genotype and weather variables using deep learning. Plos one, 16(6), p.e0252402.
9. Shahhosseini, M., Martinez-Feria, R.A., Hu, G. and Archontoulis, S.V., 2019. Maize yield and nitrate loss prediction with machine learning algorithms. Environmental Research Letters, 14(12), p.124026.
10. Shook, J., Gangopadhyay, T., Wu, L., Ganapathysubramanian, B., Sarkar, S. and Singh, A.K., 2021. Crop yield prediction integrating genotype and weather variables using deep learning. Plos one, 16(6), p.e0252402.
11. Jeong, J.H., Resop, J.P., Mueller, N.D., Fleisher, D.H., Yun, K., Butler, E.E., Timlin, D.J., Shim, K.M., Gerber, J.S., Reddy, V.R. and Kim, S.H., 2016. Random forests for global and regional crop yield predictions. PLoS One, 11(6), p.e0156571.
12. Alhnaity, B., Pearson, S., Leontidis, G. and Kollias, S., 2019, June. Using deep learning to predict plant growth and yield in greenhouse environments. In International Symposium on Advanced Technologies and Management for Innovative Greenhouses: GreenSys2019 1296 (pp. 425-432).
13. Khaki, S., Wang, L. and Archontoulis, S.V., 2020. A cnn-rnn framework for crop yield prediction. Frontiers in Plant Science, 10, p.1750.
14. Ansarifar, J., Wang, L. and Archontoulis, S.V., 2021. An interaction regression model for crop yield prediction. Scientific reports, 11(1), pp.1-14.

15. Shahhosseini, M., Hu, G. and Archontoulis, S.V., 2020. Forecasting corn yield with machine learning ensembles. Frontiers in Plant Science, 11, p.1120.
16. Shahhosseini, M., Hu, G. and Archontoulis, S.V., 2020. Forecasting corn yield with machine learning ensembles. Frontiers in Plant Science, 11, p.1120.
17. Shahhosseini, M., Hu, G., Khaki, S. and Archontoulis, S.V., 2021. Corn yield prediction with ensemble CNN-DNN. Frontiers in plant science, 12.
18. Pugnaire, F.I., Morillo, J.A., Peñuelas, J., Reich, P.B., Bardgett, R.D., Gaxiola, A., Wardle, D.A. and Van Der Putten, W.H., 2019. Climate change effects on plant-soil feedbacks and consequences for biodiversity and functioning of terrestrial ecosystems. Science advances, 5(11), p.eaaz1834.
19. Simona V. Agriculture in Nigeria - statistics and facts, https://www.statista.com/topics/6729/agriculture-in-nigeria/ [accessed January 12, 2022]
20. Doris D. S. Households participating in agricultural activities in Nigeria 2019, by type and area https://www.statista.com/statistics/1119593/households-participating-in-agricultural-activities-in-nigeria-by-type-and-area/ [accessed January 12, 2022].
21. Global climate and weather data available at https://www.worldclim.org/data/index.html [accessed November 10, 2020].
22. The maize production in Nigeria, available at https://knoema.com/data/nigeria+agriculture-indicators-production+maize?unit= [accessed January 10, 2021].
23. Soil property maps of Africa at 250 m resolution available at https://www.isric.org/projects/soil-property-maps-africa-250-m-resolution [accessed December 17, 2020].
24. Newbold, P. (1983). ARIMA model building and the time series analysis approach to forecasting. J. Forecast. 2, 23–35. doi: 10.1002/for.3980020104
25. Dickey, D. A., and Fuller, W. A. (1981). Likelihood ratio statistics for autoregressive time series with a unit root. Econometrica 49, 1057–1072. doi: 10.2307/1912517
26. Neina, D., 2019. The role of soil pH in plant nutrition and soil remediation. Applied and Environmental Soil Science, 2019.
27. Letey, J.O.H.N., 1958. Relationship between soil physical properties and crop production. In Advances in soil science (pp. 277-294). Springer, New York, NY.
28. Ray, D.K., Gerber, J.S., MacDonald, G.K. and West, P.C., 2015. Climate variation explains a third of global crop yield variability. Nature communications, 6(1), pp.1-9.
29. Kendall, MG;, 1948. Rank correlation methods. Charles Griffin Boo Series (5th ed). Okford: Oxford University Press. ISBN 978-0195208375.
30. Chen, T. and Guestrin, C., 2016, August. Xgboost: A scalable tree boosting system. In Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining (pp. 785-794).
31. Friedman, J.H., 2001. Greedy function approximation: a gradient boosting machine. Annals of statistics, pp.1189-1232.
32. Breiman, L. (2001) Random forests. Machine Learning Journal Paper, 45, 5-32
33. Ho, T.K., 1995. Random decision forests. In Proceedings of 3rd international conference on document analysis and recognition. pp. 278–282.
34. Verikas, A., Gelzinis, A. and Bacauskiene, M. (2011) Mining data with random forests: A survey and results of new tests. Pattern Recognition, 44, 330-349. doi:10.1016/j.patcog.2010.08.011
35. Haykin, S., 1994. Neural networks: a comprehensive foundation, Prentice Hall PTR.
36. Glorot, X. and Bengio, Y., 2010, March. Understanding the difficulty of training deep feedforward neural networks. In Proceedings of the thirteenth international conference on artificial intelligence and statistics (pp. 249-256). JMLR Workshop and Conference Proceedings.

37. Agarap, A.F., 2018. Deep learning using rectified linear units (relu). arXiv preprint arXiv:1803.08375.
38. Aker, J.C., 2011. Dial "A" for agriculture: a review of information and communication technologies for agricultural extension in developing countries. Agricultural economics, 42(6), pp.631-647.

Global to Multiple Local Rule-based Surrogate Model for Opening the Black Box

Shu Zhang[1], Yue Gao[1], Jun Sun[1], Shan Shan Yu[2] and Tetsu Yamamoto[2]

[1] Fujitsu R&D Center CO., Ltd., Beijing 100022, China
[2] Research Center for AI Ethics, Fujitsu Limited，Kawasaki , 211-8588, Japan
zhangshu@fujitsu.com

Abstract. Local model-agnostic explanations are the key technology to explain individual predictions of black box machine learning models, and its great advantages over directly interpretable model is the good flexibility. In recent years, although feature-based local model-agnostic explanations are widely used, it also has two problems: do not account for interactions between features and instability caused by locally randomly generated disturbance data. Focus on these issues, we propose a local explanation method, which generates rules to handle the non-linear dependences and interactions between features, utilizes these rules to select neighborhood data points to improve the stability compared with random perturbation, and then integrates the result of multiple local models to explain the black box prediction with the rules and rules' contributions. Experimental results show that our explanation outperforms the existing methods from different perspectives in terms of interpretability and fidelity.

Keywords: Explainable AI, Interpretable Model, Black-box Model.

1 Introduction

With the widespread of machine learning in many fields, there has been great interest in understanding why machine learning models make decisions. This issue is important for critical systems and decisive moments. Since, the deployed machine learning models should ensure transparency, accountability, and auditability, as the results may be catastrophic, such as in Criminal Justice [1], Medical [2], and Financial [3]. Consequently, methods of explanation have been developed, which attracted the attention from both government [4] and companies [5].

Since different users with different needs may require different explanations, there is no single approach to give an effective explanation for AI. Among these explanation methods, global and local explanations are two widely used perspectives. Global explanations attempt to know the overall behavior of the black box model, usually by employing a series of rules for explanations, such as decision rule sets [6] and rule ensembles (linear [7] and generalized linear models [8] using rule-based features). However, it is difficult to utilize an interpretable model to approximate a more complex black box model accurately. Differing therefrom, local explanation is interested

in explaining the prediction in terms of a single input and focuses on capturing the behavior of the black box model on a local region of the input space. SHAP (Shapley Additive exPlanations) assigns each feature an importance value for a particular prediction [9]. LIME (Local Interpretable Model-agnostic Explanations) utilizes a linear weighted combination of the input features to provide the explanation [10]. One issue is that these methods consider features separately, which cannot reflect the interaction between different features. When features interact with each other, this cannot be expressed as the sum of the feature effects, because the effect of one feature depends on the value of the other.

In addition to the above issue, there is another thing that affects explanation, which is how to select appropriate neighborhood data points to train the local interpretable model. Since this is directly influencing the fidelity and reliability of the explanation. Random perturbation around one instance to be explained, such as in LIME [10], may generate different explanations after each run for the same instance, which makes the explanations confusing and unreliable.

In the present paper, we focus on solving the two issues mentioned above by introducing a multiple rule-based local surrogate model. Surrogate models are model-agnostic interpretable models used to explain individual predictions of black box machine learning models. Aiming to improve the interpretability, the rules and rules' contributions are induced as the concise expression of the explanation. Rules could handle the non-linear dependences and interactions between features. To a certain extent, experts could be induced to confirm whether the exploiting rules from the data set comply with common sense and domain knowledge or not. Further, the extracted rules have been utilized to select neighborhood data points to improve the stability compared with random perturbation, and then the results of multiple local models have been integrated to explain the black box prediction.

The main contributions are as follows:

1. Global rule extraction: we induce global rules as the explanation elements to show the explanation result. Compared with independent features, rules containing a small number of statements of features, which capture non-linear dependences and interactions between features, are extracted. Meanwhile, as global rule extraction takes the distribution of the entire training data into account, the potentially important association among the features could be found. Furthermore, global information has been brought into local interpretable methods in some extent.
2. Multiple local rule-based models: we propose multiple linear models to generate the rule-based explanation for a single instance. Here, the global rules are introduced to select neighborhood data points, which ensures that these data points have the same characteristics in some respects, and then the multiple local interpretable models have been integrated to approximate to the prediction result.
3. Rules and rules' contributions as the explanations: learning from the simplicity of feature-based explanation with the form of feature and feature's weight, we show the explanation with rules and rule's weights. Furthermore, the rule's weight is changing for different instance to show their contribution, which is quite different from the global rule-based interpretable model.

The rest of the paper is structured as follows: in Section 2, related work on rule extraction and local interpretable methods are reviewed. In Section 3 we propose the multiple rule-based local surrogate model. In Section 4 the experimental results and evaluations on four different open datasets are demonstrated to assess the effects and the outputs are compared with those existing methods. Section 5 concludes the paper and presents the proposed future work.

2 Related Work

There have recently been an increasing number of studies on explanatory methods aiming to make the results of AI systems more intelligible to humans, such as reasoning through prototypes to provide useful insight into the inner workings of the network [9], or using saliency maps to highlight features in an input deemed relevant for the prediction of a learned model [11], exploiting generic problem structures in explanations for Automated Planning [12], or generating interpretable weights based on the significance of the components to further improve recognition accuracy and interpretability [13]. For the proposed multiple rule-based local surrogate model, the most closely related research involves rule-based models and local interpretable models.

Rule-based models employ a series of rules for explanation by constructing association rules with the target variables. Interpretable decision set is proposed as a joint framework for description and prediction, since the decision sets are simple, concise, and easily interpretable compared with decision trees [6]. Falling rule lists are classification models consisting of an ordered list of IF-THEN rules, provide an ability to reason about each prediction [14]. Though, the rules in the beginning of the chain of the rule list are interpretable. The additional rules become more and more difficult to understand, since it needs to understand all the rules before them, which limits the interpretability of decision lists. The most relevant RuleFit algorithm proposed by Friedman and Popescu learns a sparse linear model with the original features and several new features relating to decision rules [7]. This has filled the gap whereby the linear regression model does not account for interaction between features. Another interpretable model is the Skope-rules [15], which mainly differs in the way of choosing decision rules: semantic deduplication based on variables composing each rule as opposed to L1-based feature selection (Rulefit). Differing from the approaches (Rulefit and Skope-rules), the weights of each rule which are stable in the global explanation, are changeable according to different input instances in the proposed model. This is also the key point of local based explanation.

For local interpretable models, the most well-known model at present is LIME proposed by Ribeiro [10]. LIME explains the predictions for a classifier by learning an interpretable model locally around the prediction (Fig. 1). First, data points are generated by random perturbation around the input instance, then an interpretable linear model is learned based on this local neighborhood space. It is an open question as to what neighborhood is best to use by way of explanation. Anchor is a model-agnostic explanation method, which adopts a perturbation-based strategy to generate local explanations based on the easy-to-understand IF-THEN rule [16]. Similarly,

LORE (Local Rule-based Explanations) trains a decision tree on a set of generated artificial data points by a genetic algorithm [17]. However, there is one of the important issues of stability because of data generation, which results in getting different explanations at every run for the same instance. Focusing on this issue, Pastor and Baralis propose LACE (Local Agnostic attribute Contribution Explanation) to use K neighbors to train a local interpretable model [18]. Hall proposes K-means clustering to partition the training data set and the use thereof for training local models [5]. Zafar uses one of the clusters grouped in the training data which is nearest to the input instance as the neighborhood [19]. Differing therefrom, we propose to utilize rules to find the neighborhood. Since data points in the same rule reflect a certain degree of similarity in some aspect, which may also provide an explanation from a different perspective. Furthermore, we integrate multiple local generalized additive models to improve the approximation to the complex boundary. Instead of considering feature independently, we ascertain the interactions between different features in the form of rules.

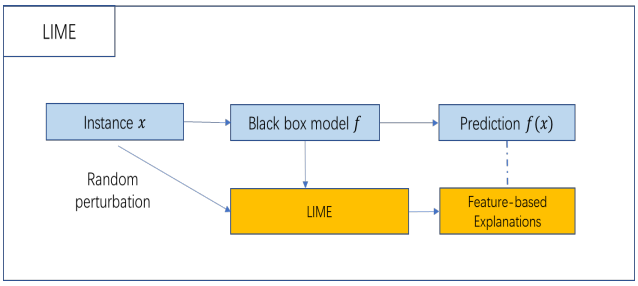


Fig. 1. The framework of LIME

3 METHODOLOGY

We consider a "black box" classifier f of predicting output target $Y \in \mathcal{Y}$(e.g., labels) using training data D with K features, where the data point $d = (d_1, \ldots, d_K)$. We then denote a single instance $x = (x_1, \ldots, x_K)$ to get a probability $f(x)$ through classifier f. Here, x is used to distinguish it from d.

This section details the proposed local surrogate interpretable model via multiple rules. We first extract global rules from training data, then for one instance to be explained, we select similar data points according to the rules of this instance matching in the entire training data. Then we propose multiple rule-based models to approximate to those black box in the local scopes with selected data points and integrate them to provide the final explanation through rules and rules' contribution. In brief, our method attempts to assign rules an important value as the explanation for a particular prediction. This process is shown in Fig. 2.

There are two challenges: one is global rule mining on training data D, and the other is generating multiple rule-based models to deduce a way in which to make black

box decisions for a single instance. As mentioned, our approach provides the rule and rule contribution-based explanations as the final explanations.

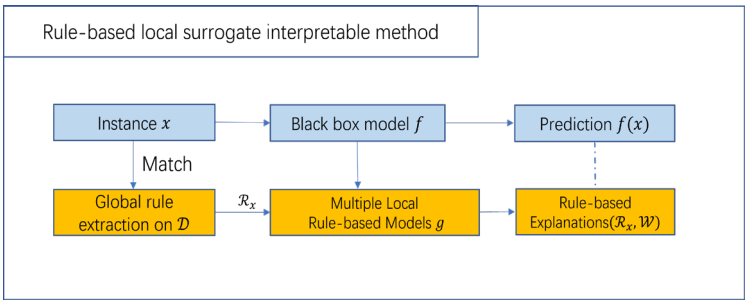


Fig. 2. The featured-based local surrogate method may ignore the interactions between features. The rule-based local surrogate interpretable method introduces global rule extraction and covers multiple local models to achieve rule-based explanations of predictions.

3.1 Global Rule Extraction

Learning rules from entire Dataset

A rule-based algorithm is one way of constructing interpretable models, which uses intrinsic interpretable algorithms such as decision tree algorithms ID3 [20], C4.5 [21] and CART [22]. The decision tree can be linearized into decision rules, where the outcome is the contents of the leaf node, and the conditions along the path form a conjunction in the IF-THEN clause. In general, the rules take the form below:

IF condition 1 and condition 2 . . . and condition M,

THEN outcome.

That is, we generate a list of rules $\mathcal{R} = \{r_1, r_2, .., r_N\}$, where r_n is composed of multiple conditions. We can also define s as the prevailing conditions, then $r_n = s_{n1} \wedge s_{n2} \wedge \ldots \wedge s_{nM}$. Here, M is decided by the depth of each tree. If it is assumed $r = s_1 \wedge s_2$, where s_1 is $x_1 > 28$ and s_2 is $x_5 = \text{“}Married\text{”}$, then this rule r is $x_1 > 28$ and $x_5 = \text{“}Married\text{”}$.

Decision trees almost always pick the root node at random and are constructed from the top-down by analyzing the interactions between features. When some features have strong correlation but are sparse in terms of the data pertaining thereto, rules may not be well extracted by use of a decision tree. As mentioned by Vapnik [23], as long as a data set for constructing the associated tree is expressive, accurate decision rules can be explored even in a smaller model space. Hence, before building the tree, we can reduce the amount of irrelevant data in advance, thus ensuring the strong correlation between the features in these data. By using the method, the clustering technique is used to divide the whole data set $\mathcal{D}$ into different subsets, which have higher similarity within one cluster and lower similarity between different clusters. Next, multiple decision trees are constructed in each cluster to generate rules, these rules are merged into a global rule list $\mathcal{R}$.

Quality measure for rules

Although the rule list $\mathcal{R}$ is obtained, there may be less data in each cluster and the quality of the rules is not guaranteed due to the uncertainty of the number of clusters, therefore, we still need to find a way of selecting high-quality rules. In this case, we not only consider how many data satisfy the rule, but also the ratio between different classes. To filter the low quality of rules, we focus on the data distribution between classes and within classes and define a new metric.

We define the data points in $\mathcal{D}$ which satisfy rule r as $cover_{\mathcal{D}}(r)$. The set of class labels in $\mathcal{D}$ is $\ell \in \{1, .., L\}$, where L is the amount of class labels. $\mathcal{D}_\ell$ is the data set in which all the data points belong to the class label ℓ. Then, the coverage ratio in class label ℓ is defined as follows:

$$\theta_\ell = \frac{\left|cover_{D_\ell}(r)\right|}{\left|cover_{D_\ell}(r)^{\complement}\right|} \tag{1}$$

Thereinto, $\complement$ is the complement set of $cover_{D_\ell}(r)$ in $\mathcal{D}_\ell$ and $|\cdot|$ is the number of set element. To compare the situation of rule coverage between and within classes, the following metrics is defined:

$$max\_ratio(\mathrm{r}) = \max_{\ell}\left(\frac{\theta_\ell}{\theta_{\neg\ell}}\right) \tag{2}$$

Thereinto, $\neg\ell$ represents the data set whose target labels are not ℓ. When the value of *max_ratio* is larger, the quality of the rule is better. We filter the rules in $\mathcal{R}$ based on the threshold of *max_ratio* and obtain the new rule list $\mathcal{R}$.

3.2 Multiple local rule-based model

Our goal is to ascertain the prediction result of a single instance through rules and rule's contributions. The rules for each instance can be easily ascertained by matching the rule list $\mathcal{R}$. So here we focus on the problem of learning the rule contributions for an instance x. First, define a rule list $\mathcal{R}_x$ where x is matched with $\mathcal{R}$. That is,

$$\mathcal{R}_x = \bigcup_{r_i \in \mathcal{R}} \{r_i | x \in cover(r_i)\} \tag{3}$$

Thereinto $i \in \{1,2, \ldots, I\}$ and $I = |\mathcal{R}_x|$, so let $r_i = s_{i1} \wedge \ldots \wedge s_{iM}$. In general, feature x_k belongs to a certain interval in each rule. Therefore, condition s_{im} can be defined as:

$$s_{im} = (x_{im}, range_{im}) \tag{4}$$

Thereinto x_{im} is a feature appearing in the condition s_{im}, which generally belongs to a contiguous interval represented by $range_{im}$ and defined by lower and upper limit. Let $z_i(x) \in \{0,1\}$ denote whether the instance x satisfies the rule r_i, as expressed by:

$$z_i(x) = \prod_{m=1}^{M} \mathbb{1}_{range_{im}} x_{im} \tag{5}$$

The indicator function $\mathbb{1}$ is 1 whenever $x_{im} \in range_{im}$ and 0 otherwise.

Since most of the classification spaces are non-linear, most of the local interpretable methods explain the classifier by learning a linear model locally around the test instance. As mentioned, the stability of LIME is poor due to the random perturbation. To ensure the stability of our explainable model, the core idea is to explain outcomes objectively through integrating multiple local interpretable models. Meanwhile, compared with the perturbation-based data generation, data under a rule tends to be more local. Formally, the data that satisfies one rule are taken as local data, which can be written as $cover_D(r_i)$. Next, we fit g_i on $cover_D(r_i)$ by using a generalized additive model. The original features d_k and their features $z_i(d)$ are combined as new features to fit a linear model:

$$g_i(d) = \alpha_0 + \sum_{k=1}^{K} \alpha_k d_k + \sum_{j=1}^{J} \beta_j z_j(d) \tag{6}$$

Through this linear model, original features and rules become a new feature and are assigned a weight under a local rule. Thereinto α_k is the weight applied to the original features and β_j is the weights of rules. Thereinto $J = I = |\mathcal{R}_x|$. If $j = i$, let $\beta_j = 0$, because under rule r_i, $z_i(d)$=1 throughout the data set $cover_D(r_i)$. Due to multicollinearity between features and rules, the proposed method uses L1 regularization. That is, we train the linear model with the ridge regression model, and the loss function is:

$$\mathcal{L}_i(f, g_i) = \sum_{d \in cover_D(r_i)} \big(f(d) - g_i(d)\big)^2 + \lambda\big(\sum_{k=1}^{K} {\alpha_k}^2 + \sum_{j=1}^{J} {\beta_j}^2\big) \tag{7}$$

Therein λ is the regularization parameter. Finally, all the g_i models and their rule weights β_{ij} are integrated as the final contribution to each rule:

$$\mathcal{W}(x) = \bigcup_j^J \frac{\sum_{i=1}^{I} Norm(\beta_{ij})}{I-1} \tag{8}$$

4 Experiments

In this section, we conducted objective experiments from different perspectives to evaluate the proposed method *MuRLoS* (*Mu*ltiple *R*ule-based *Lo*cal *S*urrogate model) on four open data sets and compare with the existing explicable method. First, we introduce the details about the datasets and the experiments setting. Then, we evaluate the rule-based explanations with two different evaluation methods focusing on the local fidelity to the black box model prediction and the extent of keeping the information from the original data. In the end, we evaluate the interpretability by simulating user understanding on the explanation.

4.1 Experiments setting and Details

We analyzed four datasets from different domains, namely Adult [1] , Bank [2] , Diabetes [3] and Churn[4]. These domains rely heavily on human decision making, and hence would benefit from the explanation of the predictive models.

- The task of Adult data set is to determine whether a person is paid more than 50,000 *per annum*.
- The classification goal of Bank data set is to predict if the client will subscribe a term deposit.
- Diabetes is an early-stage diabetes risk-prediction dataset.
- Churn aims to predict behavior to retain customers.

As a black box, here we train a neural network with two layers and 50 neurons. The aim of our experiment is not to pursue the high accuracy of the classifier, but to evaluate the interpretability of the proposed method, therefore, we can divide more test samples for explanation, with 7:3 partition ratio for training set and test set. Then we use the proposed method to provide rule-based explanations for each instance.

Setting max_ratio for global rule selection

In Section 3.1, we generate the CART trees in each cluster and merge their rules into rule list $\mathcal{R}$. Here, we use the k-means as the clustering method. And the rule list is selected by using our designed metric *max_ratio*.

To set the value of *max_ratio* to filter rules and evaluate the effectiveness of the quality measures for these selected rules, we calculate the error rate in different values of *max_ratio*. Because the outcomes through CART trees have the label ℓ for each rule, we define the $\mathcal{R}^{\ell} \subseteq \mathcal{R}$ as the rule list with label ℓ. For an instance x from training data, we define the rule list $\mathcal{R}_{\mathrm{x}}^{\ell}$ where x is matched with the rule list $\mathcal{R}_{\mathrm{x}}^{\ell}$. Here, we only consider the ratio of $\mathcal{R}_{\mathrm{x}}^{\ell}$ in $\mathcal{R}^{\ell}$ as the main factor for the prediction. Hence, the prediction for one instance x can be defined as follows:

$$\mathrm{Pred}(x) = \underset{\ell \in L}{argmax}\left(\frac{|\mathcal{R}_{\mathrm{x}}^{\ell}|}{|\mathcal{R}^{\ell}|}\right) \tag{9}$$

We adopt error rate as the metric to evaluate the rule selection. The smaller of the error rate, the performance is better. Error rate is defined according to the definition in Rulefit:

$$error\_rate = E_x \mathbb{1}[l \neq sign(\mathrm{Pred}(x))] \tag{10}$$

The Fig. 3 shows the distributions of *error_rate* in different *max_ratio* on four data sets.

1 http://archive.ics.uci.edu/ml/datasets/Adult

2 https://archive.ics.uci.edu/ml/datasets/Bank+Marketing

3 https://archive.ics.uci.edu/ml/datasets/Early+stage+diabetes+risk+prediction+dataset.

4 https://www.kaggle.com/blastchar/telco-customer-churn

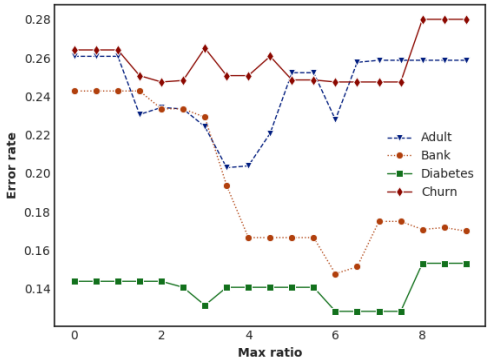


Fig. 3. The distribution of error_rate in different max_ratio on four data sets. Here the max_ratio is setting from 0.0 to 10.0, and the step is 0.5.

The *max_ratio* is useful to select rules. The larger value of the *max_ratio*, the more rules will be filtered. The scale of the rule list is decreased.

The *error_rate* is changed by different value of *max_ratio*. Based on the smallest *error_rate*, we could select one suitable value for different data set. We aim to select suitable scale of rules with high quality to explain the black box model prediction.

Details of Experiments

We list the descriptions of evaluated data, and the accuracy of the black box model in Table 1. In our experiments, we filtered the rules by the chosen value of *max_ratio* to get the final rule list $\mathcal{R}$. Here #Rules represents the total number of rules extracted from the training data. All explanations present in the paper were generated in a few seconds within a minute.

Table 1. Descriptions of data sets (Adult, Bank, Diabetes and Churn)

	#Recorders (Positive: Negative)	#Features	Accuracy on NN	#Rules
Adult	23,097:7,833	48	0.827	17
Bank	30,700:3,898	56	0.878	27
Diabetes	296:200	17	0.987	12
Churn	5,174:1,869	21	0.786	18

We give an example of our explanation result in Fig. 4 (Adult dataset), compared with by LIME and Anchor. The explanation result includes related rules and rule contributions for one input instance. These rules are self-contained and apply individually, which means that no complicated reasoning about multiple rules is needed. It makes them much more understandable to users.

Fig. 4. Example of explaining a prediction on the Adult data set.

For this example, the instance is predicted to the class who is paid less than 50,000 *per annum* by the black box model (The NN prediction is 0.43), we can see what cause him belongs to this class from the explanation. The yellow bars in the figure show positive contributions, and the blue and green bars show negative contributions of the explanation results.

Compared with LIME and Anchor, we have some similarity and difference. In details, features *Service* and *CaptialLoss* contained in the explanation of Anchor and our method, and do not appear in the explanation of LIME. Features *Married* and *CaptialGain* contained in the explanation of LIME and our method, and do not appear in the explanation of Anchor. The feature "*Married*" has been reported as an important factor in this data set, we get the same conclusion. Overall, our explanation contains more information, and the expression is more concise.

Similar with us in form, LIME gives the explanation with features and their weights. However, Anchor and LORE are rule-based explanation, they give the explanation with one decision rule and the prediction of classes. We could not know how much contribution of these features and their combination. Therefore, we choose LIME in comparison in the following experiments. In the future work, we will analyze Anchor, LORE and LACE well, hope to find a reasonable and fair way to compare with them.

4.2 Local Fidelity

For the explanation, one of the essential issues is local fidelity. The local surrogate interpretable model must be faithful to the black box model prediction. In this experiment, we use the R^2 score as the metric to evaluate the local surrogate model, with the prediction from the black box model as the ground truth. R^2 score is a relative

measure scaled between 0 and 1. The best R^2 score is 1.0. The closer the score is to 1.0, the better the performance of fidelity is to local surrogate model. We compute the mean R^2 scores with all the points in the test set for four data sets. This will indicate how good the model has fit on the whole data set. Table 2 shows the comparisons between our method and LIME for local fidelity. Our method gets higher R^2 score than LIME. It shows our method outperforms LIME by providing a better local approximation.

To verify the effectiveness of integrating multiple local interpretable models, we compare it with training one single model with the data that satisfies rules $r_i \in \mathcal{R}_x$. Our method utilizes the fitting rules $\mathcal{R}_x$ of instance x to select neighborhood data points, and then integrated with the multiple local interpretable models to approximate the prediction result. Shown in the experimental results in Table 2, our method is better than that of training one single model, which shows the effectiveness of our method. Further, the user could see more detail explanation under each local interpretable model with our method. It could be seen as giving explanation from different perspectives with different neighborhood data points.

Table 2. R^2 score on Adult, Bank, Diabetes and Churn datasets.

Datasets	MuRLoS	MuRLoS (with single linear model)	LIME
Adult	0.944	0.859	0.684
Bank	0.947	0.834	0.710
Diabetes	0.969	0.864	0.747
Churn	0.965	0.824	0.726

4.3 The Quality of Rule-based Explanations

The rule list is selected by using our designed metric max_ratio from the perspective of data distribution, but there is also a need to measure the quality of each the rule-based explanations. We want to measure the loss of feature information after the explanation. It is necessary to know whether the weight of each rule can be fitted to the same level as the original model. In other words, we wish the prediction for one instance to get no worse if the original features are substituted by the rules, and their contributions, to train a new classifier, however, we need to use the rule features of test data to train the classifier, so we cannot compare the output with that from the original black box model trained by use of the training set. Therefore, replicating the original black box model-fitting parameters, we can retrain a new model on test data as a baseline for comparison. In addition, to ensure the reliability of the test results, a cross-validation method is used.

For all data sets, the test data sets are split by 80%:20% for training and testing. Here the same neural network is used to train the classification model. According to the number of data, we conduct cross-validation on each data set and report the mean

average values of *accuracy*, *recall*, and *F1*. To ensure a fair comparison, the original features and the features of LIME are also used to perform the above experiment. The results for three methods applied to the four data sets are listed in Table 3.

The information content of proposing set of rules cannot be higher than that of black box model, therefore all the performance are lower than that of the black box model with original features shown in Table 1. The performance of our method is closer to that of using the original features and outperforms that of LIME in terms of three metrics. Although the dimension of our rule-based feature is lower than that of the original features and LIME, the effect of feature representation is better. As a result, a relatively small number of rules are generated by our method to simplify interpretation without losing information about the original features.

For the Diabetes dataset, our method and LIME have similar good performance, which is due to small scale and the presence of fewer features in this dataset. The task may be solved with interpretable models.

Table 3. Comparison of Original model, MuRLoS and LIME for the quality of rule-based explanations on Adult, Bank, Diabetes and Churn datasets.

Dataset	Method	Accuracy	Recall	F1
Adult	Original	**0.807**	0.842	**0.815**
	MuRLoS	0.806	**0.865**	0.807
	LIME	0.752	0.643	0.72
Bank	Original	**0.875**	0.905	**0.878**
	MuRLoS	0.872	**0.916**	0.876
	LIME	0.751	0.625	0.712
Diabetes	Original	0.90	0.90	0.901
	MuRLoS	**0.95**	**1.0**	**0.956**
	LIME	**0.95**	0.95	0.949
Churn	Original	0.754	0.793	0.653
	MuRLoS	**0.766**	**0.836**	**0.784**
	LIME	0.745	0.798	0.760

4.4 Simulating the Users

In this section, the interpretability is evaluated by simulating user understanding on the explanation. Given an input and an explanation, we simulate what users can predict for this input instance.

Compared with LIME, we simulate common user behaviors by summing the contribution score of rules (our method) or features (LIME) in the explanation results as the prediction value. For the instance shown in Fig. 4., we predict this instance belongs to less than 50,000 *per annum* when adding all the contributions of rules, which

is consistent with the prediction result given by the black box model. It shows that the users received the explanation of the outcome of the black box model correctly.

Accuracy is used as the metric to evaluate the performance on those test data described above. This evaluation aims to show how accurately of the rule-based explanation to humans. The results are listed in Table 4.

Table 4. Comparison results of simulating user's understanding by accuracy metric.

	Adult	Bank	Diabetes	Churn
MuRLoS	0.750	0.845	0.950	0.709
LIME	0.747	0.743	0.975	0.682

In Table 4, both LIME and MuRLoS explanation methods captured important information and showed advantages when the data set is small such as on Diabetes dataset. For the Diabetes dataset, the accuracy of LIME is higher than that of our method, which is due to the smaller scale and presence of fewer features in this dataset. On other three dataset, the results of MuRLoS are higher than those of LIME, especially on the Bank dataset with nearly up to 10%. This shows that feature-based representation performs well in terms of simple relationship data, while rule-based representation is important when there are interactions between features.

5 Conclusion

In this paper, a novel approach is proposed to explain model predictions for tabulated data. The global rule extraction presented uses global rules to select neighborhood data points. Meanwhile, rules capture non-linear dependences and interactions between features. Global rules are introduced to select neighborhood data points, and these are then integrated with multiple local interpretable models to approximate the predicted results. Compared with random perturbation, this strengthens the stability of the model to assist the generation of similar data points. Experimental results on four open datasets show that our method achieves a better performance in rule-based explanation by evaluating from different perspectives in terms of local fidelity, the extent of keeping the original information and simulating user understanding of the explanation. In future work, we will improve the fusion of multiple local interpretable models to improve the approximation of the black box model, extend our approach to deal with real large scale of data and undertake exhaustive analysis with different types of data.

References

1. Barry-Jester, A.M., Casselman, B., Goldstein, D.: The new science of sentencing., (2015).
2. Caruana, R., Lou, Y., Gehrke, J., Koch, P., Sturm, M., Elhadad, N.: Intelligible Models for HealthCare: Predicting Pneumonia Risk and Hospital 30-Day Readmission. In: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and

Data Mining, Association for Computing Machinery, pp. 1721–1730. New York, USA, (2015).
3. FICO. FICO Explainable Machine Learning Challenge, (2019).
4. Gunning, D.: Explainable artificial intelligence (xai). Technical report, Defense Advanced Research Projects Agency (DARPA), 2017.
5. Hall, P., Gill, N., Kurka, M., Phan,W.: Machine learning interpretability with H2O driverless AI, (2019).
6. Lakkaraju, H., Bach, S.H., Leskovec, J.: Interpretable Decision Sets: A Joint Framework for Description and Prediction. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1675–1684, San Francisco California USA, (2016).
7. Friedman, J.H., Popescu, B.E.: Predictive Learning via Rule Ensembles. Annals of Applied Statistics 2(3), pp. 916–954, (2008).
8. Wei, D., Dash, S., Gao, T., Günlük, O.: Generalized Linear Rule Models. In: Proceedings of the 36th International Conference on Machine Learning, Long Beach, California, pp. 6687–6696, (2019).
9. Scott M. Lundberg, Su-In, L.: A Unified Approach to Interpreting Model Predictions. In: Proceedings of the 31st International Conference on Neural Information Processing Systems, New York, USA, pp. 4768–4777, (2017).
10. Ribeiro, M.T., Singh, S., Guestrin, C.: "Why Should I Trust You?": Explaining the Predictions of Any Classifier. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Association for Computing Machinery: New York, USA, pp. 1135–1144, (2016).
11. Li, O., Liu, H., Chen, C., Rudin, C.: Deep Learning for Case-Based Reasoning Through Prototypes: A Neural Network That Explains Its Predictions. In: Thirty-Second AAAI Conference on Artificial Intelligence, Louisiana, USA, pp. 3530–3537, (2018).
12. Alan Lindsay.: Towards Exploiting Generic Problem Structures in Explanations for Automated Planning. In: Proceedings of the 10th International Conference on Knowledge Capture (K-CAP '19), New York, USA, pp. 235–238, (2019).
13. Yong, Z., Han-Zheng,W., Jia-Qi, Z., Ying, C., Rui, Y., Si-Lin, C.: Interpretable Attention Part Model for Person Re-Identification. IEEE/CAA Journal of Automatica Sinica, 41, pp.1–13, (2020).
14. Wang, F., Rudin, C.: Falling Rule Lists. In: Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics, California, USA, pp. 1013–1022, (2015).
15. Gardin, F., Gautier, R., Goix, N., Ndiaye, B., SCHERTZER, J.M.: SKOPE, (2018).
16. Ribeiro, M.T., Singh, S., Guestrin, C.: Anchors: High-Precision Model-Agnostic Explanations. In: Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans Louisiana, USA, pp. 1527–1535, (2018).
17. Guidotti, R., Monreale, A., Giannotti, F., Pedreschi, D., Ruggieri, S., Turini, F.: Factual and Counterfactual Explanations for BlackBox Decision Making. IEEE Intelligent Systems 34, pp.14–23, (2019).
18. Pastor, E., Baralis, E.: Explaining Black Box Models by Means of Local Rules. In: Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing, Association for Computing Machinery, New York, USA, pp. 510–517, (2019).
19. Zafar, M.R., Khan, N.M.: DLIME: A Deterministic Local Interpretable Model-Agnostic Explanations Approach for Computer-Aided Diagnosis Systems. In: proceeding of ACM SIGKDD Workshop on Explainable AI/ML (XAI) for Accountability, Fairness, and Transparency, ACM: Anchorage, Alaska, USA, (2019).

20. Quinlan, J.R.: Induction of Decision Trees. Machine Learning 1, pp. 81–106, Kluwer Academic Publishers, Boston, (1986).
21. Quinlan, J.R., Ed.: Programs for Machine Learning. Machine Learning 16, pp. 235-240, Morgan Kaufmann Publishers, (1994).
22. Breiman, L., Friedman, J., Olshen, R., Stone, C.J.: Classification and Regression Trees, 1st Edition, Routledge Publishers, New York, (1983).
23. Vapnik, V.N.: The Nature of Statistical Learning Theory, Springer, New York, (1995).

Modern CNNs Comparison for Fire Detection in RGB Images

Kresimir Vdovjak*, Petar Maric, Josip Balen, Ratko Grbic, Davor Damjanovic, and Matej Arlovic

Josip Juraj Strossmayer University of Osijek
Faculty of Electrical Engineering, Computer Science and Information Technology
Osijek, Kneza Trpimira 2b, 31000 Osijek, Croatia,
{kresimir.vdovjak, petar.maric, josip.balen, ratko.grbic,
davor.damjanovic, matej.arlovic}@ferit.hr

Abstract. Every year, fire causes thousands of deaths as well as billions of dollars of material damage. Prevention and early fire detection have become a topic of interest for many scientists. While there are many existing solutions such as smoke detectors, flame detectors, chemical sensors, infrared thermal cameras and many other hybrid systems, computer vision techniques that use raw RGB image as an input have emerged as fast, reliable, precise, and economical enough to be widely used with a satisfactory accuracy. For that purpose, Convolutional Neural Networks (CNNs) were considered as they can take input image from an RGB camera, learn its features and classify it as fire or non-fire. Another important thing to consider is their ability to be used on hardware with a limited amount of computational power, e.g. embedded systems. In this paper, four different versions of MobileNet, four versions of ResNet, and four versions of EfficientNet were evaluated by comparing their ability to detect fire while also taking into consideration their need for computational power. The evaluation was performed on a custom dataset that contains over 60,000 images. Overall, ResNet showed the lowest performance which was somewhat expected as it is the oldest network. MobileNets and EfficientNets showed similar performance proving themselves to be capable when used as a fire detection classifiers. Also, due to their low number of parameters and low computational need, they are suitable for use in systems with limited resources.

Keywords: convolutional neural network (CNN) · deep learning · fire detection · image classification · performance evaluation

1 Introduction

One of the most disastrous accidents that can occur in shopping centers, warehouses, factories, office buildings, schools, or any other building is fire. According to the Center of Fire Statistics [15], in 2019 there were 812,140 reported fires in residential and other buildings which is 31.6 % of all reported fires. In all these

* Corresponding author.

fires 13,938 people died which is 90.6 % of all fire-related deaths. The information was provided by 24 world countries that participate in the International Association of Fire and Rescue Service program. When we add billions of dollars in damages, the consequences are colossal, so prevention and early detection of fire is of the utmost significance.

There are many existing solutions for fire detection which include temperature detectors, smoke detectors, and even thermal cameras. Many of them have limited capabilities, cannot be used in big spaces since they have to be in the proximity of a fire source, they also can not provide the location of the source of fire, along with the size and intensity of the fire, direction of propagation, etc. Another very important factor is early fire detection, while the fire is still in the early stage or even before there is a visible flame. With the rapid growth of technology, especially in the field of digital imaging, many image and video processing techniques have been developed as they can successfully replace old types of detectors with a digital camera and some kind of computer vision software that can recognize smoke and fire. One of the most widely used and most promising techniques is using convolutional neural networks (CNNs) as they are proven to be excellent in the area of object classification. Basic idea is to use a raw RGB image as an input to the CNN without image preprocessing and without feature extraction step. CNN should automatically learn image features from input images and successfully classify them as fire or non-fire. Such fire detection systems are often used as a part of some smaller device with limited resources and often battery operated (e.g. surveillance cameras, forest fire detectors, autonomous robots, etc.) that have limited computational power and limited power supply, so the emphasis of this paper is not only on the raw performance of considered CNNs but also on their possibility to be used as a part of an embedded system.

In this paper, we compared the performance of three very popular CNNs and several of their subtypes in the task of fire classification because of the promising performance of CNNs on other publicly available datasets. Furthermore, CNN based approach to fire detection is feasible because of the cheap and widely available RGB cameras, where the main expense is in the researcher's time and computational power required to train a CNN. Traditional approaches required the use of expensive, heavy, and often large equipment which is not suitable for non-industrial applications while also requiring regular maintenance. Regardless, there are some limitations of the chosen approach in terms of the ability to detect small fire, and fire at a large distance; but the advantages of chosen apporoach overcomes the potential disadvantages. The main idea is to obtain CNN classification results on our custom fire dataset while we take into consideration their performance metrics. We compared models by evaluating confusion matrix from which we determined accuracy, precision, recall and F1-score. We also used some other important parameters such as model size, number of parameters, processing power requirements, and inference latency as they are very important when CNN in considered for embedded usage. Training was performed on Ryzen 9 5900X and RTX 3080. The rest of the hardware is presented in Table 1.

CNNs that we have used in this paper are ResNets [17], EfficientNets [27], and MobileNets [18, 19, 25].

The organization of this paper is as follows. Section 2 presents short overview of fire detection methods. In Section 3 our used dataset is presented as well as our hardware configuration and structures of used CNN models. Section 4 describes the performance of all tested models on our fire dataset. Finally, in Section 5, the conclusion and future work are given.

2 Overview of fire detection methods

As we mentioned in the introduction, there are many existing methods for fire detection. Some of them include using sensors such as temperature, smoke, and flame detectors which are becoming obsolete due to their limitations. Some of the more advanced methods for fire detection are evaluated in [16] by using chemical sensors. This kind of approach relies on the fact that chemical volatiles appear before the smoke particles. This kind of system can provide a faster response than the conventional sensors.

Another approach to consider is by using infrared thermal (IRT) cameras in combination with CNNs. The advantage of that kind of approach is its ability for early detection and prevention of fire as it can detect anomalies that still have not developed other symptoms (smoke or smell). For example, if an electrical installation is overheating due to an overload or some faulty connection, it could burst into flames after some time. By using thermal imagery, it is possible to detect an increase in temperature and react accordingly. In [20], IRT images are used for fault detection in electrical facilities. Fast Region-based CNN (Fast R-CNN), Faster R-CNN, and YOLOv3 were used as detection algorithms. The detected objects were observed through a thermal intensity area analysis (TIAA). The best accuracy was obtained by using Faster R-CNN.

During the last decade, a commonly used method for fire detection was the usage of raw RGB image processing and computer vision techniques combined with CNNs. Using IRT images and CNNs is an interesting approach to fault detection which can be further adapted for early fire detection as a part of future work. In [26], the authors used two pre-trained networks, VGG16 and ResNet-50 which they further enhanced by adding additional fully-connected layers. They tested these models on their unbalanced dataset which includes fewer fire images and thus replicating the real-world environment. Results showed improved accuracy compared to the base models but also increased training time due to the additional layers. The same approach was utilized in our paper but the stated paper used obsolete CNN architectures. Therefore we decided to evaluate newer CNN architectures in comparison to the initial ResNets. In [13], the authors proposed a novel fire detection method that consists of two steps. In the first step, a Faster R-CNN network is used to detect and localize candidate fire regions. Validation of detected fire regions was done in the second step by using analysis of spatial characteristics through Linear Dynamical Systems. Finally, to distinguish actual fire and fire-colored objects, they used VLAD encoding which further improved the performance and reduced detection errors. Obtained results were

compared with some of the most popular fire detection approaches (AlexNet, VGG16, ResNet-101) and the proposed solution retained high true positive rate while significantly reducing false positive rate since they used many fire-colored images for training. In [22], authors proposed four novel fire detection methods based on the CNN state-of-the-art object detection models, namely Faster R-CNN, R-FCN, SSD, and YOLOv3. Faster R-CNN and R-CFN belong to the two-stage object detection networks as they include region proposal network as well as classification network. In the first step, CNN takes input images and outputs region proposals. In the second step, region-based object detection CNN decides whether the fire is present or not in the proposed regions. The detection speed of two-stage networks is slower hence one-stage networks (SSD and YOLOv3) were proposed. They predict the object class by a single forward CNN. All proposed methods were evaluated on two different datasets and showed superior performance to other non-CNN based approaches. The best performance was achieved by using the YOLOv3 object detection model with 83.7 % accuracy and processing capability of 28 fps.

Often there is a need for a precise, fast, and portable solution for fire detection that can be implemented on hardware with limited computational resources, and also be available at an affordable price. In [24] authors proposed a low-cost fire detection CNN architecture based on GoogleNet since it is more suitable for implementation on FPGA and other memory-constrained hardware while retaining high classification accuracy. The proposed model consists of 100 layers with two main convolution layers, four max-pooling layers, one average pooling layer, and seven inception layers. In this work, they also used a transfer learning approach. Experiments showed excellent results in comparison with more robust models like AlexNet in terms of accuracy. In [23], authors resumed their previous work on developing cost-effective models for fire detection. They proposed a new energy-friendly and efficient CNN architecture based on SqueezeNet architecture. The proposed architecture uses smaller convolutional kernels without a dense, fully-connected layers, which helps in reducing computational requirements. The model was tested on two separate datasets and showed slightly reduced accuracy as well as reduced false positive rate. The biggest improvement was the fact that they reduced the model size from 238 MB (AlexNet) to 3 MB. The two aforementioned papers introduced novel low-cost CNN architectures suitable for embedded usage which makes them as interesting milestones in designing low storage, but high learning capacity networks.

3 Methodology

Our approach for model evaluation is to apply the same set of hyperparameters to all models in order to ensure the same training conditions among the networks. The values of used hyperparameters are presented in Table 2. Since our goal is to detect fire and to achieve the lowest possible false negative rate our models are evalutated mainly on recall and F1-score metrics. The models chosen for the analysis include MobileNets, ResNets, and EfficientNets. The reason for choosing the stated networks lays in the fact that we wanted to evaluate the

older style of CNN architecture crafting with modern state-of-the-art architecture crafting. Although MobileNets and EfficientNets share the same constituent elements, we wanted to demonstrate the EfficientNet learning capacity and scaling improvements.

3.1 Dataset

For dataset image acquisition Web scraping scripts were used because of the lack of official fire image datasets which would satisfy resolution, quantity, and variety requirements for the CNN training without inducing blur and irregularities by using various upsampling methods. Manual filtering of scraped images was required because of the grainy, low-resolution images, and images which contain conflicting or unrecognizable features even for humans. The dataset contains *challenging* images in terms of types of backgrounds, content, size of the target object, brightness/exposure, resolution, aspect ratio, computer-manipulated graphics, etc. Scraped images were complemented by the BowFire Dataset [9], Fire Dataset [3], Fire-Flame-Dataset [2], Fire Detection Dataset [4] and DFire [8]. The final dataset used for training consists of 50972 non-fire images, 7359 fire images; the validation dataset consists of 3000 non-fire images, 3000 fire images; and the test dataset of 2000 and 2000 images of non-fire and fire respectively.

A balanced training dataset may give misleading information in real-world applications because fire is a relatively rare occurrence. Hence, we do not need to create a balanced training dataset [26]. Fire images consist of various types of fire morphology, color, size, and position in the images. For networks to differentiate between non-fire images similar to fire, and real fire, many images of dawns, sunsets, neon signs, and crimson-colored objects are included. In the first row of Fig. 1 examples of challenging images of non-fire can be seen, while the second row of the aforementioned table contains examples of challenging fire images.

Fig. 1: Example of images used during the CNN training: [10], [11] [5], [6]

3.2 Hardware details

The main hardware used for training and testing consists of an x64 AMD processor and NVIDIA RTX 3080 graphic card. Fast graphic cards greatly contribute to the reduction of training times, while fast CPUs contribute to faster image queueing, loading, and preprocessing before supplying images to the model for training. Further hardware details which affect the training of machine learning the most are supplied in Table 1.

Table 1: Hardware details

Component type	**Component**
Processor	Ryzen 9 5900X
RAM	HyperX 64GB 3200 MHz
Motherboard	Gigabyte X570 Gaming X
Graphic card	RTX 3080 10 GB GDDR6X
Storage	Samsung 970 EVO Plus 1 TB

3.3 Evaluated convolutional neural networks

Tested CNNs on the custom fire dataset include four versions of MobileNets - MobileNetV1 [19], MobileNetV2 [25], MobileNetV3 [18] Small and Large variant; four versions of ResNets [17] - ResNet-18, ResNet-24, ResNet-50, ResNet-101; and four variations of EfficientNet [27] - EfficientNet-B0, EfficientNet-B1, EfficientNet-B2, and EfficientNet-B3.

As researchers tried to improve neural network performance, they modified network architectures by increasing the network depth. In theory, as we increase the number of layers in a neural network, it should get better at recognizing complex functions and features, which consequently results in better learning. That proved not to be the case as the training accuracy began to drop after a certain network depth. That was due to the problem of vanishing gradient. When the gradient is back-propagated to the earlier layers, it can converge to zero due to the repeated multiplication which results in performance degradation. So, to solve the degradation problem and still be able to use deeper networks to solve complex problems, researchers invented residual networks, one of which is ResNet [17]. The main idea was to create residual connections, i.e. identity shortcuts or skip connections. They provide alternative connections to the regular ones as shown in Fig. 2. In a training procedure, if the coefficient of a regular connection converges to zero, the residual shortcut will skip regular connection and allow the data calculated before the regular connection to be forwarded directly to the rest of the network and thus assure the integrity of the network. That process is also called identity mapping. So, if any layer hurts the performance of the networks it will be skipped.

The difference between ResNet-18 and ResNet-34, as well as ResNet-50 and ResNet-101, is just a number of used blocks that define the depth of the network while the number in the name of the ResNet subtype represents the total number of weighted layers. When considering ResNet architecture, every ResNet

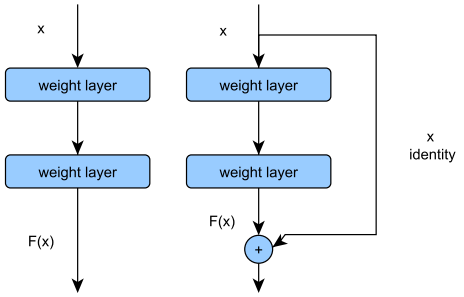


Fig. 2: Plain block (left) and residual block (right)

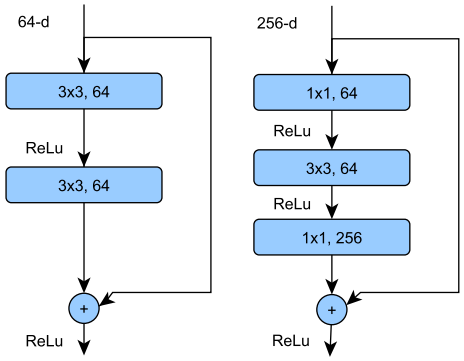


Fig. 3: Basic block (left) and bottleneck block (right) (on 56x56 feature maps)

performs initial convolution using 7x7 kernel size and max-pooling using 3x3 kernel size. Each following ResNet block is two layer deep basic block in ResNet-18 and Resnet-34, or three layer deep bottleneck block in ResNet-50 and ResNet-101 as shown in Fig. 3. The convolutional layers in the basic block have 3x3 filters and the skip connection is added to each pair of 3x3 filters while bottleneck blocks use 1x1, 3x3, and 1x1 convolutions, where 1x1 layers are used for reducing and restoring dimensions. Similar to the basic block, a skip connection is added for that three-layer block. When considering skip connections, they can be directly used when input and output are of the same dimensions. When the dimension increases, there are two possibilities. One is to use identity mapping with extra zero entries padded to compensate for increased dimension which adds no extra parameters. The second is to use a projection shortcut that matches dimension by using 1x1 convolution. Finally, all ResNets use an average pooling layer followed by a fully-connected layer.

The guiding principle for the MobileNet research team was to develop a new type of CNN that could be used on low-powered electronics with constrained computational power and energy consumption, such as mobile phones and other embedded systems. MobileNets can be used for classification, object detection, and segmentation tasks. The first iteration of the MobileNet, MobileNetV1 [19], introduces a depth-wise convolution as a vital instrument of lowering computational complexity and consequently lowering system latency. Depth-wise convolution does not require as much of the computational resources as traditional

convolution because it is applied over each channel separately, and not through all the channels at the same time. Moreover, the number of kernels convolving the image is equal to the number of channels of the input features of a certain network layer. With basic convolution, every applied kernel convolves the entire depth of the input features and the total amount of the applied kernels is arbitrarily left to the network architect to define. After applying the depth-wise convolution, an application of 1x1 kernel must be performed on its output to combine and prepare them for the following layer. MobileNetV2 [25] introduced inverted residual blocks which act as a network bottleneck, along with the skip connections which allow easier gradient backpropagation. Inverted residual blocks allow the network to transform low-level features into high-level features, i.e. more refined parts of contours of an object, while remaining computationally efficient. Heavy motivation for inverted residual blocks comes from the MobileNetV1 [19] where depth-wise convolution was being utilized. In MobileNetV2 depth-wise convolution is used to avoid using computationally expensive convolution over N input channels into one convolution per N-th channel, applying activation function and using 1x1 convolution to create output features. MobileNetV3 [18] builds on the success of its predecessor by allowing the AutoML to find the most adequate network architecture and also incorporates the squeeze-and-excitation blocks in the network architecture. Based on the [18] the authors apply MnasNet and NetAdapt algorithms from the AutoML. MnasNet is applied to find the rough network architecture which has around 80 ms of latency, and NetAdapt is applied afterward for additional performance enhancements. With their search algorithms they defined two MobileNetV3 architectures: MobileNetV3-Large suitable for more capable computing hardware; and MobileNetV3-Small for low-powered hardware. The aforementioned squeeze-and-excitation blocks improve network performance by prioritizing salient channels from the features which are sent into the block and performing feature recalibration. The structure of the squeeze-and-excitation block is the average pool layer which will have the highest value for the most salient channel, 1x1 convolution which will reduce the number of channels followed by SiLU activation function, 1x1 convolution which will output the original number of channels, and a sigmoid activation function. The output will "mark" important channels where useful information can be learned by the neural network in the deeper layers. The squeeze-and-excitation module is placed inside the inverted residual block before the final 1x1 convolution which convolves through all the channels.

The EfficientNet team's goal [27] was to find the optimal network scaling method which would deterministically improve network performance for a fixed computational cost. In other words, they could make a tradeoff between network performance and available computing resources. Previous methods of network scaling included scaling either the input image resolution, network depth, or network width. In the EfficientNet, every specified network parameter is scaled by the same constant. The authors' justification for the previous statement comes from the intuition that an increase in the image resolution requires increased depth to increase the network's receptive field, and an increased number of

Table 2: Training hyperparameters

Number of epochs	120
Training batch size	16
Validation batch size	32
Optimization algorithm	ADAM [21]
Loss function	CrossEntropy
Initial learning rate	0.001
Learning rate scheduler	ReduceLROnPlateau
- lrFactor	0.1
- lrPatience	5
- lrCooldown	4
Albumentations	inter_area interpolation horizontal flip, p=0.5 rotation, limit[-10, 10], p=0.5 scaling to range [0, 1]

channels to capture more fine-tuned features or contours. The EfficientNet has a very similar structure to the MobileNet but has more parameters because of the increased FLOPS target after applying the scaling coefficient. EfficientNet modules are taken from the various MobileNet implementations. For example, EfficientNet uses an inverted residual block from the MobileNetV2 paper [25], and squeeze-and-excitation block introduced with the MobileNetV3 paper [18]. Officially, there are eight variants of the EfficientNet depending on the amount of scaling being done across the three mentioned dimensions. Variants are in the range B0-B7 and in this paper versions from B0-B3 are evaluated.

3.4 Training parameters

To ensure the same experimental conditions for all the tested CNNs and to remove influence of different parameters on their performance, we defined training parameters and kept them fixed for all tested networks and their subtypes. Networks were implemented on Microsoft Windows 10 Pro operating system using Python 3.9 and Pytorch 1.9.0 framework [12] with CUDA 11.1 support. Training parameters are shown in Table 2. For image preprocessing the Albumentations library [1] was used with a fixed random seed in order for probabilistic preprocessing methods to apply the same transformation regardless of the computer or neural network. Also, to speed up the learning process and to achieve faster convergence, input images were normalized so that pixel values are in 0 to 1 range.

3.5 Evaluation metrics

The evaluation of selected CNNs was performed on the accuracy, precision, recall, and F1-score calculated from the entries in the confusion matrix:

- accuracy - shows how many images are predicted correctly,
- precision - shows how much positive classified images are predicted correctly, higher precision means fewer false positives,

Table 3: Model statistics

Model tier	Network name	Image resolution	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)	Lat. (ms)
L1	EfficientNet-B0	224x224	95.35	99.46	91.20	95.149	5.67
	ResNet-18	224x224	95.15	99.13	91.10	94.95	5.58
	MobileNet	224x224	94.95	99.23	90.60	94.72	6.26
L2	EfficientNet-B1	240x240	95.55	99.03	92.00	95.39	5.76
	ResNet-34	240x240	94.95	99.18	90.65	94.72	5.64
	MobileNetV2	240x240	94.82	99.23	90.35	94.58	5.11
L3	EfficientNet-B2	260x260	95.60	99.40	91.75	95.42	6.01
	ResNet-50	260x260	95.38	99.24	91.45	95.19	5.44
	MobileNetV3-Small	260x260	95.53	99.40	91.60	95.34	5.29
L4	EfficientNet-B3	300x300	95.90	99.78	92.00	95.73	6.25
	ResNet-101	300x300	93.23	99.15	87.20	92.79	5.58
	MobileNetV3-Large	300x300	95.40	99.51	91.25	95.20	5.55

- recall - shows model's ability to correctly classify positive images, higher recall means fewer false negatives,
- F1-score - harmonic mean combining precision and recall metrics,
- number of parameters - number of trainable parameters; weights that are learnt during training,
- processing power - number of performed multiply-accumulate (MAC) operations,
- model size - model size on disk in MB,
- inference latency (Lat.) - time required for an image to be passed through the model for inference in ms

Additionally, Ptflops library [7] was used to acquire total deep neural network parameter count and total multiply–accumulate operations (MAC) for a given model.

4 Results and Analysis

To compare similar models and their learning capacity, we separated models into four tiers based on the input image resolution and deep neural network complexity. In the first tier of models (L1) there are the smallest networks along with the lowest input image resolution. The L1 tier includes EfficientNet-B0, ResNet-18, and MobileNet. Higher network tiers include (EfficientNet-B1, ResNet-34, MobileNetV2), (EfficientNet-B2, ResNet-50, MobileNetV3-Small), (EfficientNet-B3, ResNet-101, MobileNetV3-Large) for L2, L3, and L4 tiers respectively. Input image resolution for each model within each tier reflects the EfficientNet input

image requirement and also facilitates the direct comparison between neural networks.

Our main evaluation results are presented in Table 3 and Fig. 4. Table 3 shows the test results of the twelve tested models on our custom fire dataset. For every model in a tier, image resolution, accuracy, precision, recall, F1-score, and inference latency are reported. In Fig. 4, number of parameters, number of operations and size on disk are presented to visually compare complexity and required computational power for each tested model. Accuracy and precision metrics are shown in the table for the sake of completeness, while the main focus for the fire detection task are recall and F1-score metrics which are also presented in Fig. 5. In the classification task, a fire event is identified with a 1, while a non-fire event is identified with a 0. Because of the purposefully imbalanced dataset, in the hope to induce the fire occurrence similar to the real-life, the main metrics of focus are recall and F1-score. False negative detection should be minimized because misclassifying real fire events has catastrophic material consequences with a high possibility of endangering human lives. Data in Table 3 displays model capabilities on completely unseen data of the test dataset which contains a total of 4000 images. The inference latency column shows the time required for an image to be passed through the model for inference and acquiring the result.

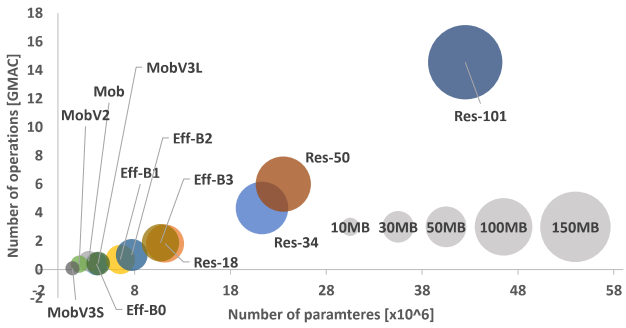


Fig. 4: Model comparison by number of operations, number of parameters and model size

Recall and F1-score graphs are the core metrics of model evaluation in this use case and are acquired during the validation phase of each epoch. Because every model was trained for 120 epochs, we can directly visualize the model behavior on the validation dataset. In this case, we argue that we can use the validation dataset for model comparison because no hyperparameters of any model were modified based on the previous performance on the validation dataset, i.e. we used a validation dataset of 6000 images as a test dataset for graphs in Fig. 5.

From Fig. 5 (5a, 5b) it can be seen that EfficientNet-B0 and ResNet-18 have a good learning start while MobileNet needs a few epochs to catch up with the learning of the other two neural networks. Advancing a few dozen epochs reveals that EfficientNet-B0 and MobileNet settled around the same value but ResNet-18 having a lower recall value during the entire validation runs. Fig. 5 (5c, 5d)

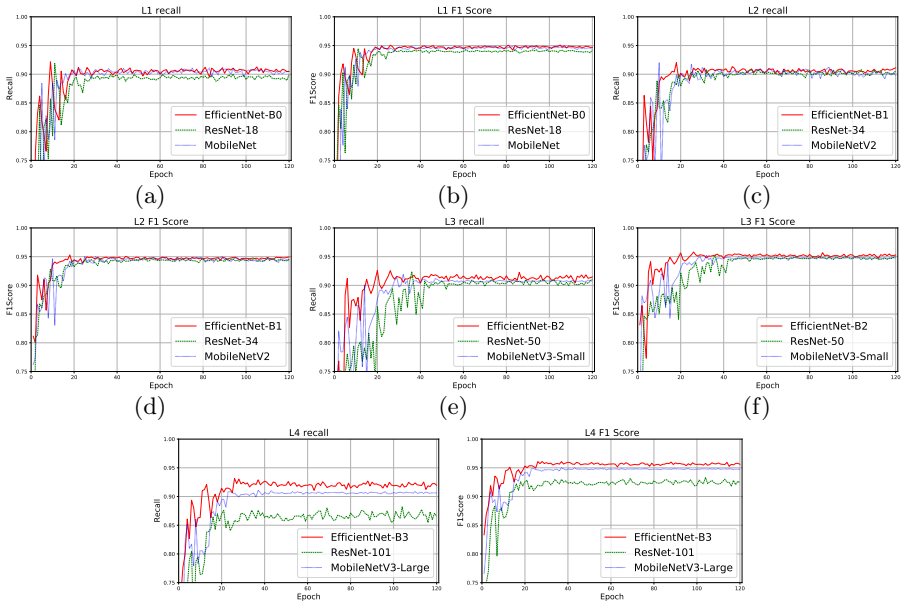


Fig. 5: L1-L4 models results after validation (recall and F1-score)

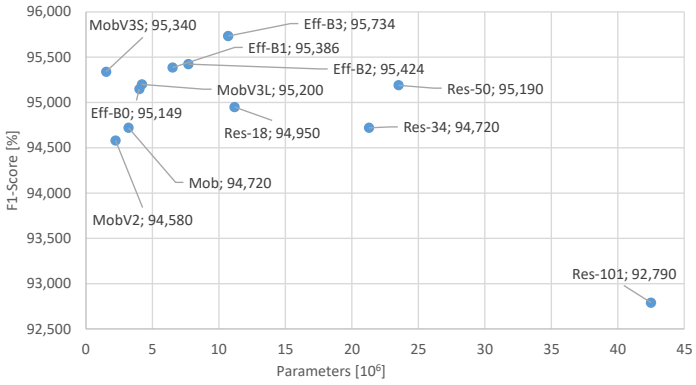


Fig. 6: Model comparison by number of parameters and corresponding F1-score

shows very similar learning characteristics between EfficientNet-B1, ResNet-34 and MobileNetV2. It can be observed that EfficientNet-B1 requires less time in contrast to the other two networks to obtain the same level of recall. Ultimately, L2 networks have very similar behavior after 50 epochs, but EfficientNet-B1 achieves a higher recall value at the end of the training. Fig 5 (5e, 5f) shows that EfficientNet-B2 has the fastest rate of learning that did not increase after the 25th epoch, and only oscillated in the following epochs. MobileNetV3-Small is somewhat slower in learning speed where it stops improving after the 28th epoch, and RestNet-50 takes the longest amount of time of 42 epochs to learn

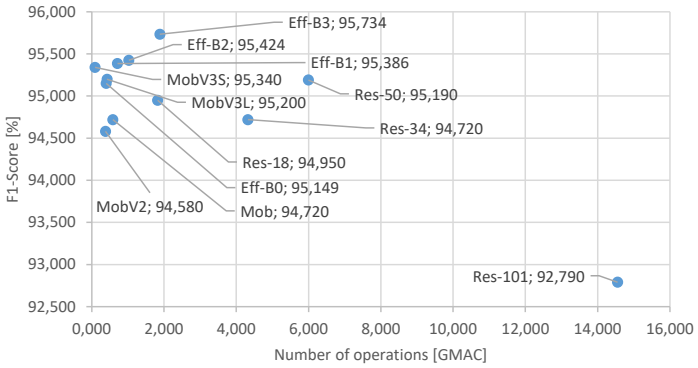


Fig. 7: Model comparison by number of operations and corresponding F1-score

the features of the custom fire dataset. EfficientNet-B2's recall remains the highest save for the few times it was equal to MobileNet-V3's recall during the entire validation run. The greatest separation between given models can be observed with the L4 tier of neural network models given by Fig. 5 (5g, 5h). The best network according to recall metric is EfficientNet-B3 closely followed by the MobileNetV3-Large. MobileNetV3-Large settles for the maximum recall value 1-2 epochs earlier than EfficientNet-B3 while having lower value. ResNet-101 falls short in comparison to the other two networks from the same tier and has a noticeably lower recall value signifying it does not have the required capacity to learn. It can be said that ResNets have become outdated in comparison with the newer network types with the likes of MobileNet and EfficientNet.
When we consider Fig. 4, it can be observed that MobileNet and EfficientNet-B0 have similar number of parameters and GMACs. When considering MobileNets and EfficientNets in the remaining evaluation tiers, it can be noted that MobileNets require lower computational power (up to 5 times less) and less storage (up to 5 times less) in comparison to the EfficientNets. Furthermore, it can be observed that ResNet models lag behind two other tested networks in terms of number of network parameters, GMACs, and model size on disk which confirms the above mentioned statement that ResNet has become obsolete. Although ResNet's results are still rather competitive, its high computational power requirements (up to 35 times more than MobileNet) and its substantially bigger size on disk (up to 10 times MobileNets size) makes it unsuitable for use on embedded systems and all other systems with limited resources. Rather surprisingly, the EfficientNets do not accomplish higher metrics in comparison to their baselines founded in MobileNets. Our conclusion for stated behavior lies in the very challenging dataset, where image resolution is highly varied along with the target content and image background used in the classification task.

Observing Fig. 6 and Fig. 7 generally we can notice close grouping of EfficientNet and MobileNet models with ResNet-101 being quite significant anomaly when we consider its number of parameters and number of operations. In Fig. 6, in our opinion, the best models in terms of a tradeoff between the number of

parameters and acquired F1-score, are MobileNetV3-Small and EfficientNet-B3; we could state the same opinion as previously mentioned models contained in the Fig. 7. In Fig. 7 the EfficientNet-B3 represents a clear advantage over other models regarding the number of operations and acquired F1-score.

Inference latency of evaluted models is presented in Table 3. All networks performed similarly, achieving 5.11 ms to 6.26 ms which, when transformed into frames per second (fps), means that networks are able to process 160 fps to 195 fps which makes them all suitable for real time applications. Another thing to consider is that this result is achieved by using high-end GPU, RTX3080, which is not suitable for embedded systems and in that case some other GPU with lower computational power would be used; thus, a much lower frame rate would be achieved.

5 Conclusion and Future Work

Every year, fire causes many deaths and billions of dollars of damage. Thus, over the last decade many new methods for fire detection emerged, especially in the field of computer vision and machine learning. This paper attempted to give a brief overview of these methods as well as various other hardware, software, and hybrid systems which could detect smoke or various chemical compounds released in the event of a fire. In the main part of the paper, the dataset acquisition process was described, along with the quantity of each class and partitioning of the dataset on the train, validation, and test subsets. Hardware specifications needed for further discussion of selected deep neural networks were also given along with the comments on how to choose optimal components. Furthermore, the main foundations of ResNet, MobileNet, and EfficientNet neural networks from their corresponding papers are presented in sufficient detail before the discussion of validation and testing results of the aforementioned models. Acquired results indicate that EfficientNets have very similar performance to MobileNets but were still managing to have slightly better model metrics. ResNets managed to maintain comparable metrics in the lower model tiers, while in the tiers with more complex models they started to show their age in comparison to EfficientNets and MobileNets. Also, when considering complexity of tested models and their need for computational power, it can be observed that newer models like MobileNets and EfficientNets achieve the same, or better results in comparison to ResNets with a lot less need for computational power which makes them suitable for embedded use cases.

Future work can be focused on the further testing of the newer CNN architectures on the custom fire dataset. The networks of interest for future testing include EfficientNetV2 [28], and ResNet-RS [14] as they represent direct competitors within the CNN machine learning field. Additionally, the focus can be directed to the modification of the existing dataset to create ground truths of fire coordinates in every image where fire occurs. With a new type of dataset, a new type of machine learning architecture can be evaluated, region-based CNNs (R-CNNs). Furthermore, the pursuit of R-CNN implementation can aid in localization of fire in RGB images.

Acknowledgment

This work is supported by the project "Research and development of autonomous robotic fire extinguisher for prevention, early detection and fire extinguishing" under the grant KK.01.2.1.02 co-financed by the European Union from the European Regional Development Fund within the Operational programme Competitiveness and Cohesion 2014-2020 of the Republic of Croatia.

Bibliography

[1] Albumentations. https://albumentations.ai/ (Jul 2021)
[2] DeepQuestAI/Fire-Smoke-Dataset. DeepQuest AI (Jul 2021)
[3] FIRE Dataset. https://kaggle.com/phylake1337/fire-dataset (Jul 2021)
[4] Fire Detection Dataset. https://kaggle.com/atulyakumar98/test-dataset (Jul 2021)
[5] Fire Image Example #1. https://bit.ly/3gFi40t (Aug 2021)
[6] Fire Image Example #2. https://bit.ly/3mE6A1e (Aug 2021)
[7] Flops-counter.pytorch/ptflops at master · sovrasov/flops-counter.pytorch. https://github.com/sovrasov/flops-counter.pytorch (Jul 2021)
[8] Gaiasd/datasets/d-fire: D-fire is an image dataset of fire and smoke occurrences. https://www.floydhub.com/gaiasd/datasets/d-fire (Jul 2021)
[9] Gbdi / bowfire-dataset — Bitbucket. https://bitbucket.org/gbdi/bowfire-dataset/src/master/ (Jul 2021)
[10] NonFire Image Example #1. https://bit.ly/3zpZyjY (Aug 2021)
[11] NonFire Image Example #2. https://bit.ly/3zoJW0m (Aug 2021)
[12] PyTorch. https://www.pytorch.org (Jul 2021)
[13] Barmpoutis, P., Dimitropoulos, K., Kaza, K., Grammalidis, N.: Fire Detection from Images Using Faster R-CNN and Multidimensional Texture Analysis. In: ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp. 8301–8305. IEEE, Brighton, United Kingdom (May 2019). https://doi.org/10.1109/ICASSP.2019.8682647
[14] Bello, I., Fedus, W., Du, X., Cubuk, E.D., Srinivas, A., Lin, T.Y., Shlens, J., Zoph, B.: Revisiting ResNets: Improved Training and Scaling Strategies. arXiv:2103.07579 [cs] (Mar 2021)
[15] Brushlinsky, N., Ahrens, M., Sokolov, S., Wagner, P.: World Fire Statistics. Tech. Rep. 26, International Association of Fire and Rescue Services (Jun 2021)
[16] Fonollosa, J., Solórzano, A., Marco, S.: Chemical Sensor Systems and Associated Algorithms for Fire Detection: A Review. Sensors **18**(2), 553 (Feb 2018). https://doi.org/10.3390/s18020553
[17] He, K., Zhang, X., Ren, S., Sun, J.: Deep Residual Learning for Image Recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 770–778 (Jun 2016). https://doi.org/10.1109/CVPR.2016.90

[18] Howard, A., Sandler, M., Chu, G., Chen, L.C., Chen, B., Tan, M., Wang, W., Zhu, Y., Pang, R., Vasudevan, V., Le, Q.V., Adam, H.: Searching for MobileNetV3. arXiv:1905.02244 [cs] (Nov 2019)
[19] Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H.: MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. arXiv:1704.04861 [cs] (Apr 2017)
[20] Kim, J.S., Choi, K.N., Kang, S.W.: Infrared Thermal Image-Based Sustainable Fault Detection for Electrical Facilities. Sustainability **13**(2), 557 (Jan 2021). https://doi.org/10.3390/su13020557
[21] Kingma, D.P., Ba, J.: Adam: A Method for Stochastic Optimization. International Conference on Learning Representations (Jan 2017)
[22] Li, P., Zhao, W.: Image fire detection algorithms based on convolutional neural networks. Case Studies in Thermal Engineering **19**, 100625 (Jun 2020). https://doi.org/10.1016/j.csite.2020.100625
[23] Muhammad, K., Ahmad, J., Lv, Z., Bellavista, P., Yang, P., Baik, S.W.: Efficient Deep CNN-Based Fire Detection and Localization in Video Surveillance Applications. IEEE Transactions on Systems, Man, and Cybernetics: Systems **49**(7), 1419–1434 (Jul 2019). https://doi.org/10.1109/TSMC.2018.2830099
[24] Muhammad, K., Ahmad, J., Mehmood, I., Rho, S., Baik, S.W.: Convolutional Neural Networks Based Fire Detection in Surveillance Videos. IEEE Access **6**, 18174–18183 (2018). https://doi.org/10.1109/ACCESS.2018.2812835
[25] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.C.: MobileNetV2: Inverted Residuals and Linear Bottlenecks. arXiv:1801.04381 [cs] (Mar 2019)
[26] Sharma, J., Granmo, O.C., Goodwin, M., Fidje, J.T.: Deep Convolutional Neural Networks for Fire Detection in Images. In: Boracchi, G., Iliadis, L., Jayne, C., Likas, A. (eds.) Engineering Applications of Neural Networks, vol. 744, pp. 183–193. Springer International Publishing, Cham (2017). https://doi.org/10.1007/978-3-319-65172-9_16
[27] Tan, M., Le, Q.V.: EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. In: Proceedings of the 36 Th International Conference on Machine Learning. vol. 97, pp. 6105–6114. Long Beach, California (2019)
[28] Tan, M., Le, Q.V.: EfficientNetV2: Smaller Models and Faster Training. arXiv:2104.00298 [cs] (Jun 2021)

Learning Adversarial Strategies

Jia Xu[1][0000−0002−2034−9029] and Michael Spece[2]

[1] University of Pennsylvania, Philadelphia, PA 19104, USA
jiaxu7@upenn.edu
[2] Carnegie Mellon University, Pittsburgh, PA 15213, USA
drenami@gmail.com

Abstract. This paper is about the relationship between regret (in online learning) and the rank of an ensemble's loss matrix Y, when Y has generated approximately adversarially. We take a learning approach to generate an approximately adversarial Y, which is realistic in terms of what one might expect to encounter in real-world problems where adversaries too must learn. We also show that this approach is more efficient than simulation and thereby obtain better empirical studies of the performance of low rank algorithms.

Keywords: online learning · regret · simulation study.

1 Introduction

The *Prediction with Experts Advice* problem is a well-known problem in prediction theory and fundamentally important in machine learning. [8] Also, this is a special case of online convex optimization. Online learning is performed in a sequence of T consecutive rounds with N experts, where at round t the learner is given a question. In each round $t = 1, \cdots, T$, the learner chooses a probability vector $p_t \in \Delta_N$, where Δ_N denotes the N-simplex, namely the set of all distributions over N experts

$$\Delta_N = \left\{ x \in R^N : \forall i, x(i) \geq 0 \wedge \sum_{i=1}^{N} x_i = 1 \right\}.$$

Then Nature chooses a loss vector $y \in [0,1]^N$, and the learner incurs the corresponding loss $p(y) = p \cdot y$. The regret is defined as follows:

$$\text{Regret}(N,T) = \sum_{t=1}^{T} p \cdot y - \min_{k \in \{1,\ldots,N\}} \sum_{t=1}^{T} y_k.$$

A learner wishes to maintain the best strategy. Therefore, if an expert performed great in previous rounds, then the expertise can be crucial guidance for learners who need advice for many reasons. The learner can rely on the chosen expert's advice and decide for each round. This strategy is known as Follow the Leader and can be interpreted as a form of empirical risk minimization.

As described in [11], there are two types of online learning settings: stochastic and adversarial. We undertake the adversarial setting that is less well understood and the focus of this paper.

1.1 Online Convex Optimization and Programming

In online convex optimization, an online player iteratively makes decisions. At the time of each decision, the outcomes associated with the choices are unknown to the player. Therefore, prediction from experts' advice is a special case of Online Convex Optimization. The overall target is to minimize a continuous and convex function over a convex subset of Euclidean space.

Convex programming is a generalization of linear programming with many applications to machine learning. A new general framework was presented for convex optimization over matrix factorizations, where every Frank-Wolfe iteration will consist of a low-rank update [9]. The researchers also discussed the broad application areas of this approach.

Many algorithms can be used in online convex optimization, including Follow the Regularized Leader (FTRL), Exponentially Weighted Averaging (EWA, also known as Hedge), and Projected Gradient Descent (PGD).

EWA is named and defined in [2]. Its fundamental idea is to weigh experts based on their performance. More precisely, an expert with relatively high loss will receive an exponentially decreasing proportion of weight. The update rule is

$$p_j^{t+1} = \frac{p_j e^{-\eta y_j}}{\sum_{i=1}^{N} p_i e^{-\eta y_i}}. \tag{1}$$

This algorithm can be modified in many ways. One of these modifications is called the doubling trick, in which the learning rate η becomes adaptive over exponentially increasing time epochs [2]. Consequently, T need not be known in advance. We consider another horizon-adaptive algorithm in Section II and the generic EWA. These algorithms will be studied through simulation.

FTRL modifies Follow the Leader by adding a regularization term R ([6]), which can provide stability to the algorithm. In this case, it can help ensure the algorithm does not overfit. The standard form for FTRL for the problem at hand reduces to

$$p = \text{argmin}_{x \in \Delta_N} \sum_{t'=1}^{t-1} x \cdot y^{t'} + R(x). \tag{2}$$

OMD is a transformation of FTRL. Moreover, its update rule is naturally interpreted as a gradient operation.

Other algorithm variants include lazy updating (e.g. [6]).

1.2 Algorithms for Gradient Descent Optimization

Gradient descent (GD) is the simplest and oldest optimization method. It is an iterative method in the optimization procedure that proceeds in iterations, each improving the objective value [8].

Gradient descent algorithms for optimization have been gaining popularity [12]. Some common gradient descent algorithms are Gradient Descent (GD), Batch Gradient Descent (BGD), Stochastic Gradient Descent (SGD), Mini-batch Gradient Descent (MBGD), Momentum, Nesterov Accelerated Gradient (NAG), Adagrad, Adadelta, RMSprop, Adaptive Moment Estimation (Adam), AdaMax, Nadam.

Since metioned previously, the study[14] shows that gradient descent algorithm has been proved fruitful on this problem, as the average will approach zero. They proposed greedy projection, a gradient descent-based technique for general convex functions. They applied the Greedy Projection to select an arbitrary x_1 $\in$ F and a sequence of learning rates η_1, η_2, $\ldots \in R^+$. In time step t (t = 0, 1, $\cdots$.), after receiving a cost function, select the next vector x_{t+1} according to:

$$x^{t+1} = P(x^t - \eta_t \nabla c^t(x^t)). \tag{3}$$

A unified framework was also developed to analyze the behavior of projected gradient descent in application to low rank estimation problems[3]. In their work, they also used the projected gradient descent updates, that is

$$F^{t+1} = F^t - \eta^t \nabla \widetilde{L_n}(F^t). \tag{4}$$

The researchers in [10] used the gradient descent method earning in multi-layer linear networks leads to minimum-rank solutions. This shows the essential application of gradient descent, and this method can minimize the rank.

The rank of the covariance matrix of the codes is implicitly minimized by using the gradient descent learning in multi-layer linear networks leads to minimum-rank solutions [10].

This study will focus on implementing the Exponentially Weighted Averaging and Projected Gradient (subgradient) Descent method in an adversarial setting.

1.3 Restart Approaches

These have been studied in various fields in online learning for years. One of the most famous applications is the adaptive regret for fixed shares given by [1]. It uses the restart approach for prediction under mix-loss by implementing the Follow the Leading History presented by [7].

[5] applied the restart approach to EWA under branching experts. [4] also applied the restart approach to EWA so that both the number of effective experts and the step parameter can be set dynamically.

[11] provided the foregoing review of low rank learning and found various settings, including sub-settings of [6], where $O\left(\sqrt{rT}\right)$ is achievable.

1.4 Numerical Analysis

Because theory has given an incomplete characterization of the regret or even when it is $O\left(\sqrt{rT}\right)$, [13] resorted to the numerical analysis of the regret for particular algorithms by simulating adversarial strategies.

1.5 Our Contribution

We take a slightly different approach in that we attempt to learn the adversarial strategy. Depending on our findings, this appears more efficient.

2 Nature's Problem

Nature will learn over K T-horizon games. Its loss in each trial k is therefore

$$\sum_{t=1}^{T} p^{k,t} \cdot y^{k,t} - \min_{n \in \{1,\dots,N\}} \sum_{t=1}^{T} y_n^{k,t}.$$

To enforce a rank restriction on Nature, constrain $y^{k,t}$ to be of the form $U^k \ell^{t,k}$ where $U^k \in \mathbf{R}^{N \times r}$ and $\ell^{t,k} \in [0,1]^r$.

2.1 Learning Algorithm: Projected Subgradient Descent

The learning algorithm employed by Nature will be projected online subgradient descent based on [14]. There are dependencies on trial, time, and expert. For each trial, Nature determines its strategy overall time. Then one iterates over time to compute the EWA response. For comparison, note projected gradient descent is both a form of proximal gradient descent and a form of OMD; the latter has already been discussed as a form of regularized FTL.

By the calculus of subdifferentials, the subgradient (concerning y, the variable under Nature's control) is (omitting k)

$$-\left(\sum_{t=1}^{T} \nabla p^t \cdot y^t - \sum_{t=1}^{T} \nabla y_{n_*}^t\right)$$

for any n_* such that $\sum_{t=1}^{T} y_{n_*}^t = \min_{n' \in \{1,\dots,N\}} \sum_{t=1}^{T} y_{n'}^t$.

$$\begin{aligned} \frac{\partial p^t \cdot y^t}{\partial y_n^t} &= p^t \cdot \frac{\partial y^t}{\partial y_n^t} \\ &= p_n^t, \end{aligned}$$

so

$$\sum_{t=1}^{T} \nabla p^t \cdot y^t = p$$

$$\frac{\partial y^t_{n_*}}{\partial y^t_{n_*}} = 1,$$

is the matrix which is 0 everywhere but the n_*th row where it is 1.

To simplify the projection, we consider substituting the constraints and redoing the subgradients calculations with respect to (U, ℓ) for a fixed k (omitted).

$$\frac{\partial p^t \cdot U\ell^t}{\partial U_{n,d}} = p^t_n \ell^t_d,$$

so

$$\sum_{t=1}^{T} \nabla p^t \cdot U\ell^t$$

$$\frac{\partial U^t_{n_*}}{\partial y^t_{n_*}} = 1,$$

For greater simplicity, rather than project U, L to guarantee $U^k\ell^{t,k} \in [0,1]^N$, we project the products $U^k\ell^{t,k}$ themselves into $[0,1]^N$. We consider the algorithm (Algorithm 1) the projected subgradient descent algorithm.

Algorithm 1: Projected Subgradient Descent (Nature)

Data: Input U,L, probablity vector p (the t th column of Y is called y^t,the k th rows and the t th column of Y is called y^t_k)

Result: Get the regret

1. $p^* = 0(k = 1, \ldots, N)$
2. Find p^* which minimizes $\sum (UL)^t \cdot p^*$.
3. $\text{subgradU}_{ij} = (x^*_i - x_i t) \cdot L_{tj}, i = 1, \ldots, N, j = 1, \ldots, r$
4. $\text{subgradL}_{tj} = U_i^{j\,\mathrm{T}} \cdot x_{it}, t = 1, \ldots, T, j = 1, \ldots, r$
5. $U = U - \text{subgradU} \cdot \dfrac{\text{initial step size}}{\sqrt{\text{trial}}}, \text{trial} = 1, \ldots, \text{nTrials}$
6. $L = L - \text{subgradL} \cdot \dfrac{\text{initial step size}}{\sqrt{\text{trial}}}, \text{trial} = 1, \ldots, \text{nTrials}$

3 Learning Algorithms Under Simulation Study

The algorithm (Algorithm 2) we consider is a horizon-adaptive version of Hedge, which also appears in [2].

Algorithm 2: Hedge Algorithm

Data: Input N,T, loss matrix Y (the t th column of Y is called y^t,the k th rows and the t th column of Y is called y_k^t)

Result: Get the regret

1. $p_1^1 = 1$ $p_k^1 = 0 (k = 2, \ldots, N)$
2. $p_k^t = \frac{e^{-\eta \sum_{t'=1}^{t-1} y_k^{t'}}}{\sum_{k'=1}^{N} e^{-\eta \sum_{t'=1}^{t-1} y_{k'}^{t'}}} (\eta = \sqrt{\frac{\log(N)}{t} \times 2}, t = 2, \ldots, T, k = 1, \ldots, N)$
3. $(N, T) = \sum_{t=1}^{T} p^t \cdot y^t - \min_{k \in \{1,\ldots,N\}} \sum_{t=1}^{T} y_k^t$

We now combine the Hedge algorithm with Projected Subgradient Descent Algorithm (Nature), the new algorithm.

Algorithm 3: New Algorithm

Data: Input nTrials, N, T and Rank
Result: Worst regret for each rank after nTrials
initialization;
while *r in range (1,Rank + 1)* **do**
    U, L = numpy.empty((T, r)) ;
    L[:, 0:r] = numpy.random.binomial(1, .5, size=(T, r));
    U[:, 0:r] = numpy.random.binomial(1, .5, size=(N, r));
    Y = numpy.dot(U, numpy.transpose(L));
    p = Hedge(N, T, Y);
    while *k in range (1,nTrials)* **do**
        U, L, Y = Nature(U, L, p, T, k);
        p, cumLoss = Hedge(N, T, Y);
        Reg = regret(Y, cumLoss);
        WorstRegret[k] = Reg.max();
    end
    WorstRegret = WorstRegret.max();
end

4 Simulation

In the simulations of the loss matrix, each vector component follows a binomial distribution.

In the following experiments, we fix $N = 20$, $T = 20$ if we do not especially mention.

4.1 Generate The Loss Matrix

The Algorithm 4 is to generate the matrices of a not high rank r.

Algorithm 4: How to generate the loss matrix, which has N rows and T columns of a not high rank r (compared with the $\min(N, T)$)

Data: Input a distribution D and a certain rank r
Result: Generate the loss matrix of a not high rank r (compared with the $\min(N, T)$)

1. Generate three matrices: A, B, C. A is a matrix of N rows and T columns, and its each vector component obeys a binomial distribution.
   B is a matrix of N rows and N columns, and its each vector component obeys a binomial distribution.
   C is a matrix of N rows and N columns, and its first r diagonal numbers are 1, other numbers are 0.
2. $Y = B \times C \times A$. And if Y's element is greater than 1, make it equal to 1; if Y's element is smaller than 0, make it equal to 0
3. Make a judgment: if the Y' s rank equals to r, then use it as the loss matrix; if not, abandon it.

However, it is not efficient for higher rank r (compared with the $\min(N, T)$): C is close to the identity matrix of $\min(N, T)$ dimensions. So Y is close to the $B \times A$, and it is very easy for its elements to be bigger than 1. Then according to Algorithm 3, the Y will be a matrix whose most elements are all one and rank is lower than expected. Therefore, it is tough to get the worst regrets of high rank r in Algorithm 3.

So we consider another way (Algorithm 5):

Algorithm 5: How to generate the loss matrix, which has N rows and T columns of a certain rank r (high)

Data: Use binomial distribution and a certain rank r
Result: Generate the loss matrix of a high rank

1. Generate a matrix: Y. And Y is a matrix of N rows and T columns,and its each element obeys a binomial distribution.
2. The Y is used as the loss matrix.

Algorithm 5 also has its disadvantage: when using this way to generate the matrices of low rank r (compared with the $\min(N, T)$), it is tough to get the matrices that we want. The matrix of rank r must have $(N - r)$ linear dependent rows and $(T - r)$ linear dependent columns. When N, T are much bigger than r, this can be very hard.

Therefore, in the following experiments, we use Algorithm 3 to generate the matrices of relatively low rank r and Algorithm 4 to generate the matrices of relatively high rank r. (when $N = T = 20$, we think r is relatively high if $r \geq 18$); (when $N = T = 50$, we think r is relatively high if $r \geq 42$)

4.2 $N = T = 20$ $(p = 0.5)$

Fig. 1 shows the relationship between the worst regret and the loss matrices' rank r with the new algorithm.

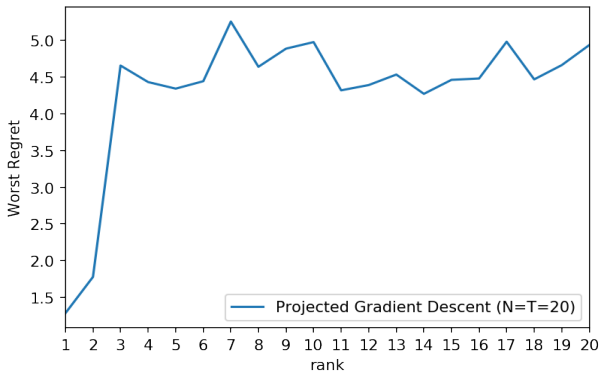


Fig. 1: 100000 trials of Projected Gradient Descent Algorithm ($p = 0.5$), $N = 20, T = 20$

Furthermore, in Experiment B, we will use only the learning algorithm (Algorithm 1) and call this algorithm – naive simulation in our following figures. The new learning algorithm, combined with the hedge and projected subgradient descent algorithm, is exploited to compute the regrets. Moreover, we will use projected gradient descent in our following figures to annotate this new algorithm.

Fig. 2 shows the worst regret comparison between the naive simulation and the projected gradient descent algorithm.

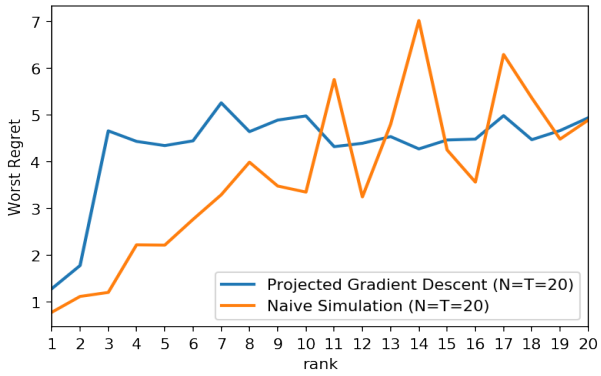


Fig. 2: Comparison of Projected Gradient Descent and Naive Simulation ($p = 0.5$), $N = 20, T = 20$, $nTrials = 100000$

In Experiment B, we fix the $N = T = 20$, and we use Algorithm 2 with binomial distribution ($p = 0.5$) to generate the loss matrices of certain rank r, when $r \leq 17$; and use Algorithm 3 with the distribution $D =$ binomial distribution ($p = 0.5$) to generate the loss matrices of rank r, when $r \geq 18$. For each certain rank r ($r \in \{1...\min(N, T)\}$), we do 100000 independent trials.

From the Fig. 1, we can find three results:

1. The worst regret can decrease when the rank increases.
2. The worst regret experienced a sharp increase between rank 1 and 4.
3. The curve has a rapid decline when the rank begins to be relatively high.

About Fig. 1, it was expected that the worst regret would keep increasing when the rank increases. However, our finding is inconsistent with that. To find if our finding is just because the number of the trial is not significant enough to get the accurate worst regrets, we do Experiment C.

Furthermore, from the Fig. 2, we can find three results:

1. The worst regret computed by the new algorithm is greater than naive simulation in the first ten ranks.
2. The worst regret computed by the new algorithm is more stable than naive simulation, especially in the last ten.
3. The curve has a rapid decline when the rank begins to be relatively high.

4.3 The Accurate Worst Regret Of Small N, T

In Experiment C, we also use naive simulation to compute the regrets and fix the $N = T = 5$.

However, this time, we generate all the possible matrices with five rows and five columns, and these matrices' elements are chosen from set $\{0, 1\}$. In this case, we can get the accurate worst regret.

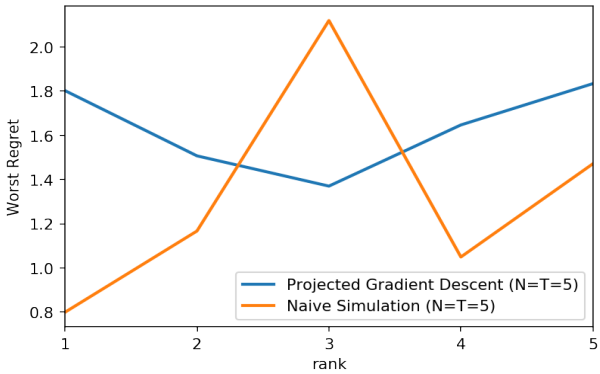


Fig. 3: All the result of binomial distribution, $N = 5, T = 5$

Fig. 3 is the picture describing the relationship between the worst regret and the loss matrices' rank r for both algorithms. Then we compute the regret of all the possible loss matrices of rank r, and choose the maximum value of them as the worst regret of the rank r. So what we get is the accurate worst regret.

From the picture and data, we can see that the worst regret can decrease when the rank increases. So the result in the Experiment B (the worst regret can decrease when the rank increases) is reasonable.

4.4 For Larger N And T

Fig. 4 is the plot describing the relationship between the worst regret and the loss matrices' rank r.

In Experiment D, we fix the $N = T = 50$, and we use Algorithm 3 with the binomial distribution ($p = 0.5$) to generate the loss matrices of rank r. For each certain rank r ($r \in \{1...\min(N, T)\}$), we do 100,000 independent trials. In addition, we only show the first 20 ranks' worst regret in order to compare with the previous experiments and also be more time-saving.

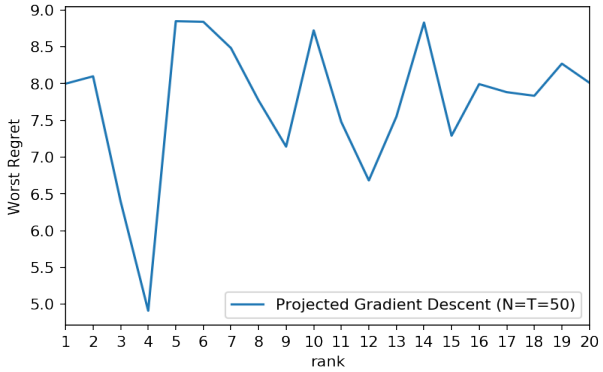


Fig. 4: 100000 trials for each rank of binomial distribution ($N = T = 50$)

From Fig. 4 and Fig. 5, we find the last three results in Experiment B are also true, which proves that Experiment B's findings can not only be applied to particular N and T but also to bigger N and T.

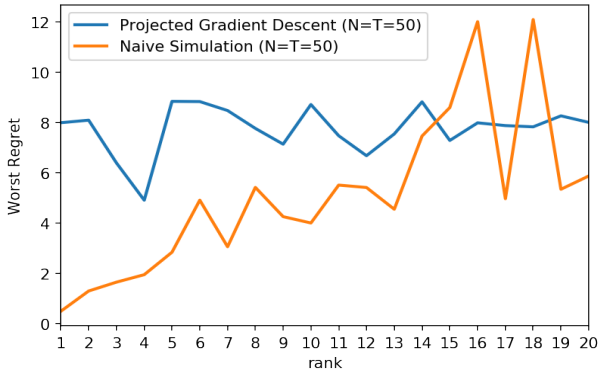


Fig. 5: 100000 trials for each rank of binomial distribution ($N = T = 50$)

Using the Hedge algorithm only to generate the loss matrices of rank r, and using the new algorithm to upgrade the gradients of the loss matrices before the

Hedge to generate the loss matrices of rank r is quite different. We also find that the new algorithm operates more stable than the hedge algorithm alone.

It is reasonable that the points, where r is nearly 20, are getting smaller. Because when the r ($r \leq 42$) is getting close to 20, it is harder and harder to generate the loss matrices of rank r using the Algorithm 3. So the regret will decrease. We present all the results in Fig. 6.

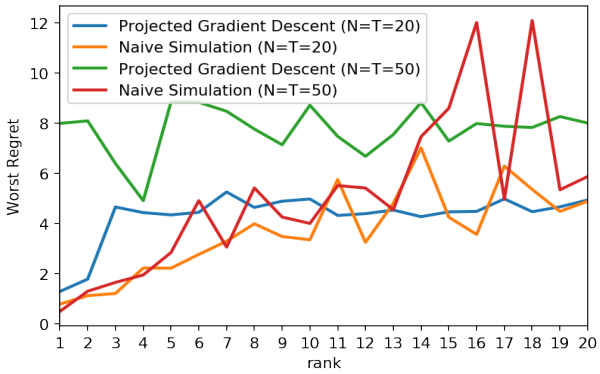


Fig. 6: 100000 trials for each rank of binomial distribution ($N = T = 50$)

5 Conclusion

Yang[13] suggested the difficulty of proving accurate upper bounds is not as much in algorithm design as it is in uncovering how algorithms take advantage of coincidental structure. Here we have undertaken the study of such behaviors using simulation, offering an alternative to the brute force sampling method employed in Yang.

In comparing our plots to Yang, we observed higher worst-case regret. This suggests our method has better iteration complexity. However, future studies of these two methods should consider the time cost of each iteration.

Given our results are experimental, theoretical justification or other independent verification of them is warranted.

References

1. Adamskiy, D., Koolen, M.W., Chernov, A., Vovk, V.: A closer look at adaptive regret. J. Mach. Learn. Res. **17**(1), 706–726 (Jan 2016), http://dl.acm.org/citation.cfm?id=2946645.2946668
2. Cesa-Bianchi, N., Lugosi, G.: itshapePrediction, Learning, and Games
3. Chen, Y.: Fast low-rank estimation by projected gradient descent : General statistical and algorithmic guarantees pp. 1–63 (2015)
4. Cohen, A., Mannor, S.: Online learning with many experts. CoRR **abs/1702.07870** (2017), http://arxiv.org/abs/1702.07870

5. Gofer, E., Cesa-Bianchi, N., Gentile, C., Mansour, Y.: Regret minimization for branching experts. In: Shalev-Shwartz, S., Steinwart, I. (eds.) Proceedings of the 26th Annual Conference on Learning Theory. Proceedings of Machine Learning Research, vol. 30, pp. 618–638. PMLR, Princeton, NJ, USA (Jun 2013), http://proceedings.mlr.press/v30/Gofer13.html
6. Hazan, E., Koren, T., Livni, R., Mansour, Y.: Online learning with low rank experts. CoRR **abs/1603.06352** (2016), http://arxiv.org/abs/1603.06352
7. Hazan, E., Seshadhri, C.: Efficient learning algorithms for changing environments. vol. 382, p. 50 (01 2009). https://doi.org/10.1145/1553374.1553425
8. Hazan, E.: Introduction to Online Convex Optimization. arXiv (2019). https://doi.org/10.1561/2400000013
9. Jaggi, M.: Revisiting Frank-Wolfe : Projection-Free Sparse Convex Optimization **28** (2013)
10. Jing, L., Zbontar, J., Lecun, Y.: Implicit Rank-Minimizing Autoencoder (NeurIPS) (2020)
11. Liu, W., Spece, M., Jia, S.: Fast learning and low rank experts: Novel adaptive methods. SSRN (Oct 2019), https://ssrn.com/abstract=3484205, available at https://ssrn.com/abstract=3484205
12. Ruder, S.: An overview of gradient descent optimization pp. 1–14 (2016)
13. Yang, W., Spece, M.: Implicit Adaptation to Low Rank Structure in Online Learning **11**(3) (2021). https://doi.org/10.18178/ijmlc.2021.11.5.1058
14. Zinkevich, M.: Online Convex Programming and Generalized Infinitesimal Gradient Ascent. Proceedings, Twentieth International Conference on Machine Learning **2**, 928–935 (2003)

Traffic Surveillance Video Analytics: A Concise Survey

Hadi Ghahremannezhad[1], Chengjun Liu[1], and Hang Shi[2]

[1] New Jersey Institute of Technology
Department of Computer Science
Newark, NJ 07102 USA
[2] Innovative AI Technologies
Newark, NJ 07201, USA

Abstract. With the evolution of intelligent transportation systems, video data obtained from traffic scenes has become one of the most beneficial sources of information. The recent advances in the fields of pattern recognition and computer vision along with the developments in hardware capabilities have attracted many researchers to attempt to apply innovative algorithms to analyze traffic video data. In this survey, a comprehensive analysis of the main tasks in automatic traffic surveillance systems is presented. These tasks include foreground segmentation, object detection and classification, tracking, region-of-interest determination, and incident detection. The state-of-the-art techniques developed for each step of traffic video analytics are highlighted and the major challenges are discussed along with the future scope and directions.

Keywords: Video Analytics, Traffic Surveillance, Object Detection, Tracking, Foreground Segmentation, Shadow Removal

1 Introduction

There is an inevitable need to monitor road traffic in order to improve the efficiency and safety of road transportation. During the 1980s, the technological advancements in information, control, communication, GPS, computers, microprocessors, and sensing, has led to the development of intelligent transportation systems (ITS) [3]. Intelligent transportation systems have broad applications in emergency vehicle notification, automatic road enforcement, setting various speed limits, collision avoidance, car navigation, traffic signal control, parking guidance, and automatic number plate recognition. The performance of these systems depends on the accuracy and efficiency of data collection and processing.

When it comes to making a cost-efficient choice among various sensor technologies, camera sensors are preferred in most cases due to simple installation steps, rich visual information, and a relatively vast area of coverage provided by each device. In general, there are two types of camera sensors used in traffic management applications, namely, traffic enforcement cameras and traffic surveillance cameras. The former category refers to the cameras that are mounted

beside or over a road and capture high-resolution pictures to detect motoring offenses and enforce traffic rules, whereas the latter refers to the remotely controllable cameras that are typically mounted on high poles and street lights and capture live lower-resolution videos with the goal of observing the traffic and detecting incidents.

The data received from the traffic surveillance cameras are used for vehicle counting, object classification, speed estimation, and detecting events such as congestion, stopped vehicles, wrong-way vehicles, and accidents. Manually processing the huge amount of video data by human operators is a tedious and impractical task and the results are not always reliable. With the advances in fields of artificial intelligence, pattern recognition, and computer vision it is of no surprise that automated techniques in intelligent video analytics have gained a lot of attention in traffic management systems [1]. In this survey, we discuss the core components of intelligent video analytics and their applications and challenges in processing traffic video data. The remainder of the paper is structured as follows: Section 2 contains a brief overview of the main steps and the various approaches applied in vision-based traffic monitoring systems. Section 3 outlines the main challenges faced by the intelligent video analytics techniques. Finally, further discussions and conclusions are given in Section 4.

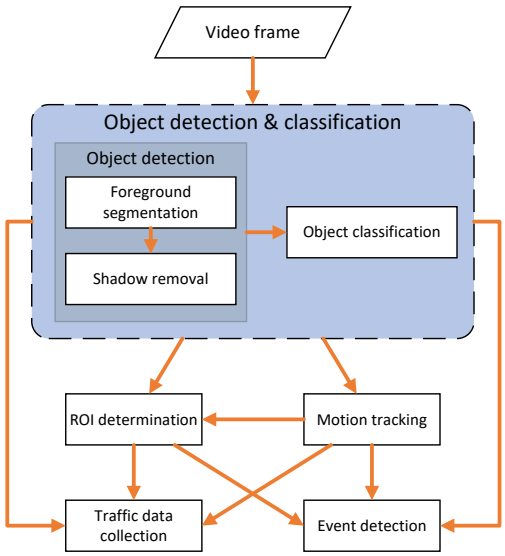


Fig. 1. General flowchart of an intelligent traffic video analytics framework.

(a) Original frame (b) MOG foreground (c) GFM foreground

Fig. 2. The foreground masks extracted using the MOG method and the GFM method, respectively. The GFM method extracts both the moving vehicles (blue) and the stopped vehicles (red) clearly [19].

2 Main steps in intelligent traffic video analytics

Traffic flow monitoring based on video analytics has been one of the main applications of computer vision. There are several steps taken towards developing a fast and reliable system in order to process traffic videos. Traffic surveillance cameras, commonly powered by electricity or solar planes, are installed along major roads in urban areas, such as highways, motorways, freeways, and intersections to provide consistent live imagery data. The live video is sent to a monitoring center and is processed by applying intelligent video analytic techniques in order to recognize traffic patterns, collect traffic flow parameters, and detect incidents and anomalies. In terms of intelligent video analytics, several steps are taken to process the raw video data and generate useful information in real-time. The major steps after camera calibration and pre-processing are composed of vehicle detection, classification, and tracking, region of interest determination, and incident detection [42]. Figure. 1 illustrates the general steps in an intelligent traffic video analytics framework. This section contains an overview of the different approaches used for each step.

2.1 Vehicle detection

Traffic surveillance cameras overlook roads, highways, and urban traffic environments and the objects of interest are all road users including different types of moving vehicles, cyclists, and pedestrians. Among different object detection techniques, background modeling has been proven to be an efficient approach to segment the foreground and obtain the location of moving objects in traffic videos [43]. Background subtraction methods are generally categorized into five groups, namely, basic, statistical, fuzzy, neural networks, and non-parametric methods [37]. Statistical methods have shown notable performance and efficiency which has made these methods most popular in real-time surveillance applications. In most used statistical approaches the background is estimated using frame averaging [10], single Gaussian [64], or a mixture of Gaussian distributions [55].

(a) (b) (c)

Fig. 3. The foreground mask after removing shadows. (a) Original video frame. (b) Ground truth. (c) the results of the Ghahremannezhad et al. [20] method.

Statistical methods based on Gaussian distribution have been the most used pixel-based approaches for modeling the background in videos due to the compromise between their computational cost and performance. Initially, each pixel was modeled using a single Gaussian distribution [64]. Later, to remove the effect of noise, camera jitter, and background texture, Gaussian Mixture Model (GMM) was proposed and each pixel was modeled by a mixture of K Gaussian distributions [55]. The GMM method was further improved upon by efficient parameter updating in adaptive GMM (AGMM) [68] and other pattern recognition techniques [50,9,48]. Here, we review the GMM method as the most commonly used approach in the applications of traffic surveillance videos.

The GMM method is the most popular technique in background subtraction due to its good trade-off between performance and efficiency. The Global Foreground Modeling method (GFM) [48] is one of the best GMM-based methods in foreground segmentation. This study has increased the dimensionality of the feature vector in order to improve the discriminatory power of the feature vectors. One of the advantages of the GFM method isthe ability to continuously detect the foreground objects even after they are stopped. This is because a separate global model is trained for the foreground objects and the GFM model will maintain an accurate Gaussian distribution to model the foreground pixels. Figure. 2 shows and example of the foreground segmentation applied on a real traffic video data.

Shadow removal One of the main challenges of foreground segmentation in traffic videos is the cast shadows of vehicles which has negative effects on further video analytics tasks, such as vehicle tracking and classification. Most foreground detection methods classify the shadows caused by moving objects as foreground due to the similarities in the motion patterns among the shadows and the moving objects. Many studies have attempted to detect the moving cast shadows and remove them from the foreground class using various approaches, such as applying color information [21], applying texture information [22], or a combination of low-level features [49].

(a) video frame (b) ground truth (c) ROI

Fig. 4. Examples of road extraction in traffic videos [16].

The state-of-the-art method in removing shadows from traffic surveillance videos is presented by Ghahremannezhad et al. [20], which considers the physics-based properties of reflection to identify potential outdoor shadows and applies a new region-based shadow detection method along with statistical modeling for robust shadow detection. An illumination invariant feature is used as input for a clustering-based segmentation method where each segment is further classified into shadow/object classes. In addition to this, six-dimensional feature vectors are constructed for each pixel and modeled by a mixture of Gaussian distributions to classify the pixels into shadows or objects. Figure. 3 illustrates an example of moving cast shadow removal by applying the method introduced in [20].

2.2 Region of interest determination

A region of interest (ROI), is a sample within a dataset identified for a particular purpose [6]. In the case of image and video analytics, the region of interest refers to a subspace of the image or video frame that indicates the region of the main focus. In the case of traffic surveillance video, the region of interest usually refers to the road area and its proximity as the objects of interest are the road users. In most traffic video applications the ROI is selected manually by human agents which are required to be performed for every video during system installation.

Many studies have attempted to determine the region of interest automatically by proposing various approaches which are mostly referred to as road detection, lane detection, road segmentation, road boundary detection, vanishing point detection, or ROI determination. One group of road detection methods tend to segment the road region in traffic surveillance videos by applying low-level features [16,18,47,17]. Deep convolutional neural networks (DCNNs) have also been applied for various segmentation tasks including road segmentation [33]. Ghahremannezhad et al. [18] propose a novel Adaptive Bidirectional Detection (ABD) of region-of-interest (ROI) to segment, the roads with bidirectional traffic flow into two regions of interest automatically. In other studies, Ghahremannezhad et al. [16,17] propose a model for fusing various discriminating features to extract the road region reliably and in real-time. In [17], the initial road samples are obtained in a similar way after localizing the moving vehicles

by applying the GFM method. In Figure. 4 some examples of road segmentation can be seen by applying the method introduced in [16].

2.3 Vehicle classification

Vehicle classification is one of the main functionalities of traffic monitoring systems and has broad applications in intelligent transportation systems and smart cities. A high-performance classification system that is able to classify the vehicles into different types accurately, can help the traffic monitoring systems with effective operations and transportation planning.

Many studies have used methods based on deep neural networks (DCNNs) in order to extract high-level and effective features for the task of vehicle classification [26,14,40]. Faruque et al. [14] apply two representative DCNN-based methods, namely Faster R-CNN [41] and YOLO [40] to classify vehicles in three datasets constructed from videos that are provided by the New Jersey Department of Transportation (NJDOT). The vehicles are classified into six types as defined by the Federal Highway Association (FHWA): bike, car, truck, van, bus, and trailer. According to the evaluations, YOLO has performed faster and with higher precision in comparison to Faster RCNN.

2.4 Vehicle tracking

The next significant step after detecting the objects in traffic videos is to associate the locations of the objects at each frame with their corresponding location across multiple consecutive frames in order to track the motion of vehicles and other road users. Vehicle tracking in traffic videos has been categorized in a number of different approaches based on features extracted from the data and vehicle representation [12] which are briefly reviewed.

Region-based tracking The most-used approaches for tracking multiple vehicles in traffic videos are based on regions, also called blobs, each of which is defined to be a connected component of the image where the pixels of each component have common properties, such as similar intensities, color, texture, and temporal features. In this group of methods each region is associated with one vehicle and is tracked across the frames by using various clues, such as appearance, motion, and geometrical properties. As an example of blob-tracking methods, we can refer to the approach applied by Chang et al. [8] where the foreground blobs at each frame are associated with the closest blobs of the previous frame in terms of the Euclidean distances between blob centroids. Region-based tracking methods are computationally efficient and work well for traffic surveillance videos, especially the videos captured from the highways. Nevertheless, these methods have poor performance in case of traffic congestion, vehicle occlusion, and cluttered background as the regions merge and the vehicles cannot be distinguished.

Contour-based tracking Another group of multiple object tracking methods tends to represent the objects using their contours or so-called boundaries which are updated dynamically at each frame. These methods have a lower computational overload in comparison to the region-based methods as they provide a more efficient description of the vehicles and other road users. However, the same problems of the region-based methods caused by vehicle occlusions and merging of the contours result in drastic drops in the performance of tracking. These sorts of problems are usually dealt with by active track grouping strategies and heuristic update policies [44].

Feature-based tracking This group of tracking methods uses various features to represent the vehicles instead of connected components. Various representative features, such as corners, general shape, lines, and speeded up robust features (SURF) are extracted at each frame and matched over the sequence of frames. The performance of this group of tracking methods relies on the discriminatory power of the selected features. Most feature-based tracking methods [44,15,35,24] work well under different illumination conditions, however, the partial vehicle occlusions can have negative effects on them.

Model-based tracking In this group of tracking methods a model is projected onto the image and matched across the frames based on the vehicle motion. In model-based methods, [11,4,38] the trajectories, models, and the geometrical poses of the vehicles are recoverable and estimated more accurately compared to other approaches. For example, Danescu et al. [11] model the vehicles as three-dimensional cuboids that are generated by grouping the three-dimensional points which are obtained using stereo-vision. The tracking is performed by matching the measurements and the models which are in turn done by intersecting the rectangles in the bird's-eye view and a corner-by-corner matching in the image space.

2.5 Incident detection

Besides collecting useful traffic data one of the main functionalities of traffic monitoring systems is to detect or predict various traffic incidents in real-time in order to take appropriate actions upon discovering an unusual event. Along with various sensors which the road networks are equipped with, surveillance cameras provide useful and continuous information, such as speed, occupancy, and traffic flow. This information can be used for Automatic Incident Detection (AID) in order to notify the operators in the control center who are on the alert for unusual traffic conditions. Here, we review some of the most common traffic incidents and the various approaches attempted to detect the events automatically by applying different algorithms.

Stopped vehicle detection Stopped vehicles in the middle or on the shoulders of roads and highways pose a great deal of danger to the drivers and passengers

and are one the main reasons for traffic accidents. According to the AAA Foundation for Traffic Safety, approximately 12 percent of interstate highway deaths were due to accidents on the road shoulders. There have been a number of studies with the specific goal of detecting stopped vehicles in traffic surveillance videos. For example, Alpatov et al. [2] apply a simple background subtraction technique for detecting stopped vehicles by subtracting the earliest background estimations from the current background estimation. Several other studies [53,58,5,51,48], which are mostly referred to as abandoned object detection or stationary object detection methods, have also attempted to propose various approaches to detect the objects such as vehicles continuously after they stop moving.

Wrong-Way vehicle detection Wrong-way driving (WWD), also known as counterflow driving, refers to the act of driving a motor vehicle against the predefined correct direction of traffic. One of the applications of intelligent traffic video analytics is detecting wrong-way vehicles in real-time based on various vehicle detection and tracking techniques. There are a number of studies that have focused on the task of wrong-way vehicle detection [39,52,23]. Ha et al. [23] implement a system that combines background subtraction with an optical flow algorithm to analyze and learn the motion orientation of each lane and detect the vehicles moving in the wrong direction. Rahman et al. [39] propose an automatic wrong-way vehicle detection system for on-road surveillance cameras.

Traffic congestion detection One of the main purposes of traffic management systems is to detect traffic congestion, otherwise known as traffic jams, in order to analyze the information for better planning. Road traffic congestion refers to a condition in transport where the number of vehicles exceeds a certain limit based on the capacity of the road lanes which results in longer trip times and increased vehicular queuing. Many studies have attempted to detect traffic congestion [30,62,7,31] in videos which helps the traffic monitoring centers with useful information about traffic jams. Ke et al. [30] present a multi-step method for road congestion detection that consists of foreground density estimation based on gray-level co-occurrence matrix, speed detection based on Lucas-Kanade optical flow algorithm with pyramid implementation, background modeling based on the Gaussian mixture model, and applying a convolutional neural network (CNN) for vehicle detection from the candidate foreground objects.

Traffic accident detection Traffic accidents, also referred to as vehicle collisions or car crashes occur when a vehicle collides with other vehicles, pedestrians, animals, or stationary objects on or around the road. There are several types of traffic accidents, including rear-end, side-impact, head-on collisions, vehicle rollovers, or single-car accidents. There have been several studies about different approaches addressing the issue of detecting traffic accidents immediately after their occurrence [66,27,65,28,32,59,19].

Ghahremannezhad et al. [19] propose a novel and real-time single-vehicle traffic accident detection framework which is composed of an automatic region-

of-interest detection method, a new traffic direction estimation method, and a first-order logic (FOL) traffic accident detection approach. The traffic region is detected automatically by applying the global foreground modeling (GFM) method and based on the general flow of the traffic. Then, the traffic direction is estimated based on blob-tracking in order to identify the vehicles that make rapid changes of direction. Finally, the traffic accidents are detected using the rules of the first-order logic (FOL) decision-making system. This method is capable of detecting single-vehicle run-off-road crashes in real-time which is one of the most important types of traffic accidents, especially on roads and highways.

Traffic anomaly detection Aside from the studies that are focused on detecting a specific event, there are many studies that attempt to detect abnormal incidents as anomaly [61,56,57,36,13,46]. Wei et al. [61] consider stopped vehicles as anomalous behavior and introduce a novel method based on the mixture of Gaussian (MOG) background subtraction method in order to obtain the background which contains the stopped vehicles. The Faster R-CNN method is applied on the subtracted background in order to detect the stopped vehicles and report them as anomalies. Shi et al. [46] present a novel anomalous driving detection method by applying a new multiple object tracking (MOT) method, and a novel Gaussian local velocity (GLV) modeling method to model the moving patterns.

Example Automatic Traffic Incidents Detection Systems The Smart Traffic Video Analytics (STVA) [34] systems for automatic traffic incidents detection and automated traffic volume counting are featured on the NJDOT Technology Transfer website[1]. Some examples of automatic traffic incidents detection and automated traffic volume counting using hundreds of real traffic videos from New Jersey Department of Transportation (NJDOT) and South Korea are shown on the following websites[2, 3]. Table 1 shows some of the other existing visual traffic surveillance systems along with their main components.

3 Main challenges in intelligent traffic video analytics

In order to have a reliable vision-based traffic monitoring system the applied algorithms are supposed to be generalizable, accurate, and responsive, all at the same time [67]. The traffic video streams should be analyzed automatically and in real-time in order to provide the operators at traffic monitoring centers with useful information about the collected traffic data, such as volume, speed, and vehicle types, as well as to detect incidents and create alerts for different events, such as stopped vehicles, slow-speed, wrong-way vehicles, pedestrians, and traffic accidents. Here, we briefly discuss the main challenges in intelligent traffic video analytics systems.

[1] www.njdottechtransfer.net/2021/11/08/automated-video-analytics
[2] https://web.njit.edu/~cliu/NJDOT/DEMOS.html
[3] https://iaitusa.com

Table 1. Some of the recent 2D visual traffic surveillance systems and their major components.

Ref.	Year	Detection	Tracking	Incidents
Wang et al. [60]	2019	YOLO	Kalman filter [29]	Stopped vehicle, Traffic anomaly
Ijjina et al. [27]	2019	Mask RCNN	Centroid tracking	Accident
Singh et al. [54]	2019	Background subtraction	N/A	Accident
Seong et al. [45]	2019	YOLOV2	Kalman filter and IOU	N/A
Huang et al. [25]	2020	YOLO	Deep SORT [63]	Near-Accident
Ghahremannezhad et al. [19]	2020	Background subtraction	Blob tracking	Single-Vehicle accident
Hang et al. [46]	2021	Background subtraction	MOT	Stopped vehicle, Traffic anomaly
Liu et al. [34]	2021	Background subtraction	Blob tracking	Stopped vehicle, Traffic congestion

3.1 Performance and reliability

In order to be able to trust an intelligent traffic video analytics system with automatic data processing and incident detection, the operators at the traffic monitoring center tend to set the system at a sensitive level to minimize undetected events. A traffic operation center may house a large number of traffic video feeds simultaneously and an accurate intelligent video analytics system can help lighten the workload of operators to a great extent. Nevertheless, if the intelligent video analytics framework fails to collect sound and reliable data and detect all incidents correctly without too many false alarms, it can become a liability and increase the manual work. The intelligent traffic video analytics systems are deployed based on the existing infrastructures and the provided input video data has relatively low resolution and frame-rate which makes it even more challenging to design effective algorithms.

3.2 Flexibility and versatility

Traffic videos are continuously captured during day and night in many locations and can vary in weather condition, illumination, resolution, and frame-rate. Some CCTV (Closed Circuit Television) cameras provide the functionality of adjusting Pan, Tilt, and Zoom (PTZ) for the operator which means a wide range of possible changes in camera viewing angle and distance. All these various possibilities in visual properties among traffic surveillance videos, make it challenging to construct a generalizable framework for performing the tasks of intelligent traffic video analytics. There are factors such as illumination and weather conditions that cannot be controlled by the operators, and the intelligent video analytics system is expected to be able to adapt well in various situations.

3.3 Efficiency and responsiveness

One of the main factors in designing intelligent traffic video analytics systems is the processing time which is expected to be short enough to be able to process multiple video frames at each second. Real-time responsiveness is a required property for traffic video analytics applications due to the continuous stream of video frames that should be processed without delay. The computational capacity of the underlying platform, the video quality, and resolution, the number of video frames captured at each time interval, and compliance with the specified cost-efficiency policies are the key points of consideration in defining the limits for the complexity of the designed algorithms.

4 CONCLUSIONS

In this survey, the developments and challenges in intelligent traffic video analytics are reviewed briefly. After a general introduction of the benefits of intelligent traffic video analytics in traffic management systems, the main steps are discussed along with some of the most popular methods applied at each step. First, some of the representative studies about object detection and classification are summarized. Various methods proposed for the tasks of foreground segmentation, shadow removal, and vehicle classification are described that are applied to detect and classify various road users in traffic videos. Then the benefits of defining a region of interest are discussed along with a number of studies that introduce different algorithms to detect the region of interest automatically.

Since incident detection is one of the main goals of intelligent traffic video analytics systems besides collecting traffic data, a section is dedicated to providing a brief introduction to the studies that have attempted to detect common traffic events. Afterward, the major challenges in intelligent traffic video analytics are briefly outlined. The imperfection of the current technologies in dealing with various challenging factors in video analytics reveals that additional efforts are needed for further improving the performance and generalization capability of the current traffic video analytics systems, which renders an important field of study with promising research opportunities.

References

1. Abdulrahim, K., Salam, R.A.: Traffic surveillance: A review of vision based vehicle detection, recognition and tracking. International journal of applied engineering research **11**(1), 713–726 (2016)
2. Alpatov, B.A., Ershov, M.D.: Real-time stopped vehicle detection based on smart camera. In: 2017 6th Mediterranean Conference on Embedded Computing (MECO). pp. 1–4. IEEE (2017)
3. Auer, A., Feese, S., Lockwood, S., Hamilton, B.A.: History of intelligent transportation systems. Tech. rep., United States. Department of Transportation. Intelligent Transportation ... (2016)

4. Barth, A.: Vehicle tracking and motion estimation based on stereo vision sequences. Ph.D. thesis, Inst. für Geodäsie und Geoinformation (2013)
5. Bayona, Á., SanMiguel, J.C., Martínez, J.M.: Stationary foreground detection using background subtraction and temporal difference in video surveillance. In: 2010 IEEE International Conference on Image Processing. pp. 4657–4660. IEEE (2010)
6. Brinkmann, R.: The art and science of digital compositing: Techniques for visual effects, animation and motion graphics. Morgan Kaufmann (2008)
7. Chakraborty, P., Adu-Gyamfi, Y.O., Poddar, S., Ahsani, V., Sharma, A., Sarkar, S.: Traffic congestion detection from camera images using deep convolution neural networks. Transportation Research Record **2672**(45), 222–231 (2018)
8. Chang, F., Chen, C.J., Lu, C.J.: A linear-time component-labeling algorithm using contour tracing technique. computer vision and image understanding **93**(2), 206–220 (2004)
9. Chien, S.Y., Chan, W.K., Tseng, Y.H., Chen, H.Y.: Video object segmentation and tracking framework with improved threshold decision and diffusion distance. IEEE transactions on circuits and systems for video technology **23**(6), 921–934 (2013)
10. Chowdhury, A., Cho, S.j., Chong, U.P.: A background subtraction method using color information in the frame averaging process. In: Proceedings of 2011 6th International Forum on Strategic Technology. vol. 2, pp. 1275–1279. IEEE (2011)
11. Danescu, R., Nedevschi, S., Meinecke, M.M., Graf, T.: Stereovision based vehicle tracking in urban traffic environments. In: 2007 IEEE Intelligent Transportation Systems Conference. pp. 400–404. IEEE (2007)
12. Datondji, S.R.E., Dupuis, Y., Subirats, P., Vasseur, P.: A survey of vision-based traffic monitoring of road intersections. IEEE transactions on intelligent transportation systems **17**(10), 2681–2698 (2016)
13. Doshi, K., Yilmaz, Y.: Fast unsupervised anomaly detection in traffic videos. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops. pp. 624–625 (2020)
14. Faruque, M.O., Ghahremannezhad, H., Liu, C.: Vehicle classification in video using deep learning. In: Machine Learning and Data Mining in Pattern Recognition, MLDM. pp. 117–131. ibai publishing, Leipzig (2019)
15. Furuya, T., Taylor, C.J.: Road intersection monitoring from video with large perspective deformation. Ph.D. thesis, University of Pennsylvania (2014)
16. Ghahremannezhad, H., Shi, H., Liu, C.: Robust road region extraction in video under various illumination and weather conditions. In: 2020 IEEE 4th International Conference on Image Processing, Applications and Systems (IPAS). pp. 186–191. IEEE (2020)
17. Ghahremannezhad, H., Shi, H., Liu, C.: Automatic road detection in traffic videos. In: 2020 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking (ISPA/BDCloud/SocialCom/SustainCom). pp. 777–784. IEEE (2020)
18. Ghahremannezhad, H., Shi, H., Liu, C.: A new adaptive bidirectional region-of-interest detection method for intelligent traffic video analysis. In: 2020 IEEE Third International Conference on Artificial Intelligence and Knowledge Engineering (AIKE). pp. 17–24. IEEE (2020)
19. Ghahremannezhad, H., Shi, H., Liu, C.: A real time accident detection framework for traffic video analysis. In: Machine Learning and Data Mining in Pattern Recognition, MLDM. pp. 77–92. ibai publishing, Leipzig (2020)

20. Ghahremannezhad, H., Shi, H., Liu, C.: A new online approach for moving cast shadow suppression in traffic videos. In: 2021 IEEE International Intelligent Transportation Systems Conference (ITSC). pp. 3034–3039. IEEE (2021)
21. Gomes, V., Barcellos, P., Scharcanski, J.: Stochastic shadow detection using a hypergraph partitioning approach. Pattern Recognition **63**, 30–44 (2017)
22. Guo, R., Dai, Q., Hoiem, D.: Paired regions for shadow detection and removal. IEEE transactions on pattern analysis and machine intelligence **35**(12), 2956–2967 (2012)
23. Ha, S.V.U., Pham, L.H., Tran, H.M., Thanh, P.H.: Improved optical flow estimation in wrong way vehicle detection. Journal of Information Assurance & Security **9**(7) (2014)
24. Hsieh, J.W., Yu, S.H., Chen, Y.S., Hu, W.F.: Automatic traffic surveillance system for vehicle tracking and classification. IEEE Transactions on Intelligent Transportation Systems **7**(2), 175–187 (2006)
25. Huang, X., He, P., Rangarajan, A., Ranka, S.: Intelligent intersection: Two-stream convolutional networks for real-time near-accident detection in traffic video. ACM Transactions on Spatial Algorithms and Systems (TSAS) **6**(2), 1–28 (2020)
26. Huttunen, H., Yancheshmeh, F.S., Chen, K.: Car type recognition with deep neural networks. In: 2016 IEEE intelligent vehicles symposium (IV). pp. 1115–1120. IEEE (2016)
27. Ijjina, E.P., Chand, D., Gupta, S., Goutham, K.: Computer vision-based accident detection in traffic surveillance. In: 2019 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT). pp. 1–6. IEEE (2019)
28. Jiansheng, F., et al.: Vision-based real-time traffic accident detection. In: Proceeding of the 11th World Congress on Intelligent Control and Automation. pp. 1035–1038. IEEE (2014)
29. Kalman, R.E.: A new approach to linear filtering and prediction problems (1960)
30. Ke, X., Shi, L., Guo, W., Chen, D.: Multi-dimensional traffic congestion detection based on fusion of visual features and convolutional neural network. IEEE Transactions on Intelligent Transportation Systems **20**(6), 2157–2170 (2018)
31. Kurniawan, J., Syahra, S.G., Dewa, C.K., et al.: Traffic congestion detection: learning from cctv monitoring images using convolutional neural network. Procedia computer science **144**, 291–297 (2018)
32. Lee, I.J.: An accident detection system on highway using vehicle tracking trace. In: ICTC 2011. pp. 716–721. IEEE (2011)
33. Li, H., Chen, Y., Zhang, Q., Zhao, D.: Bifnet: Bidirectional fusion network for road segmentation. arXiv preprint arXiv:2004.08582 (2020)
34. Liu, G., Shi, H., Kiani, A., Khreishah, A., Lee, J., Ansari, N., Liu, C., Yousef, M.M.: Smart traffic monitoring system using computer vision and edge computing. IEEE Transactions on Intelligent Transportation Systems (2021)
35. Ma, X., Grimson, W.E.L.: Edge-based rich representation for vehicle classification. In: Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1. vol. 2, pp. 1185–1192. IEEE (2005)
36. Nguyen, K.T., Dinh, D.T., Do, M.N., Tran, M.T.: Anomaly detection in traffic surveillance videos with gan-based future frame prediction. In: Proceedings of the 2020 International Conference on Multimedia Retrieval. pp. 457–463 (2020)
37. Nurhadiyatna, A., Jatmiko, W., Hardjono, B., Wibisono, A., Sina, I., Mursanto, P.: Background subtraction using gaussian mixture model enhanced by hole filling algorithm (gmmhf). In: 2013 IEEE international conference on systems, man, and cybernetics. pp. 4006–4011. IEEE (2013)

38. Ottlik, A., Nagel, H.H.: Initialization of model-based vehicle tracking in video sequences of inner-city intersections. International Journal of Computer Vision **80**(2), 211–225 (2008)
39. Rahman, Z., Ami, A.M., Ullah, M.A.: A real-time wrong-way vehicle detection based on yolo and centroid tracking. In: 2020 IEEE Region 10 Symposium (TENSYMP). pp. 916–920. IEEE (2020)
40. Redmon, J., Farhadi, A.: Yolov3: An incremental improvement. arXiv preprint arXiv:1804.02767 (2018)
41. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. Advances in neural information processing systems **28**, 91–99 (2015)
42. Rodríguez, T., García, N.: An adaptive, real-time, traffic monitoring system. Machine Vision and Applications **21**(4), 555–576 (2010)
43. Saran, K., Sreelekha, G.: Traffic video surveillance: Vehicle detection and classification. In: 2015 International Conference on Control Communication & Computing India (ICCC). pp. 516–521. IEEE (2015)
44. Saunier, N., Sayed, T.: A feature-based tracking algorithm for vehicles in intersections. In: The 3rd Canadian Conference on Computer and Robot Vision (CRV'06). pp. 59–59. IEEE (2006)
45. Seong, S., Song, J., Yoon, D., Kim, J., Choi, J.: Determination of vehicle trajectory through optimization of vehicle bounding boxes using a convolutional neural network. Sensors **19**(19), 4263 (2019)
46. Shi, H., Ghahremannezhad, H., Liu, C.: Anomalous driving detection for traffic surveillance video analysis. In: 2021 IEEE International Conference on Imaging Systems and Techniques (IST). pp. 1–6. IEEE (2021)
47. Shi, H., Ghahremannezhadand, H., Liu, C.: A statistical modeling method for road recognition in traffic video analytics. In: 2020 11th IEEE International Conference on Cognitive Infocommunications (CogInfoCom). pp. 000097–000102. IEEE (2020)
48. Shi, H., Liu, C.: A new foreground segmentation method for video analysis in different color spaces. In: 2018 24th International Conference on Pattern Recognition (ICPR). pp. 2899–2904. IEEE (2018)
49. Shi, H., Liu, C.: Moving cast shadow detection in video based on new chromatic criteria and statistical modeling. In: 2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA). pp. 196–201. IEEE (2019)
50. Shimada, A., Arita, D., Taniguchi, R.i.: Dynamic control of adaptive mixture-of-gaussians background model. In: 2006 IEEE International Conference on Video and Signal Based Surveillance. pp. 5–5. IEEE (2006)
51. Shyam, D., Kot, A., Athalye, C.: Abandoned object detection using pixel-based finite state machine and single shot multibox detector. In: 2018 IEEE International Conference on Multimedia and Expo (ICME). pp. 1–6. IEEE (2018)
52. Simpson, S.A., et al.: Wrong-way vehicle detection: proof of concept. Tech. rep., Arizona. Dept. of Transportation (2013)
53. Singh, A., Sawan, S., Hanmandlu, M., Madasu, V.K., Lovell, B.C.: An abandoned object detection system based on dual background segmentation. In: 2009 Sixth IEEE International Conference on Advanced Video and Signal Based Surveillance. pp. 352–357. IEEE (2009)
54. Singh, D., Mohan, C.K.: Deep spatio-temporal representation for detection of road accidents using stacked autoencoder. IEEE Transactions on Intelligent Transportation Systems **20**(3), 879–887 (2018)

55. Stauffer, C., Grimson, W.E.L.: Adaptive background mixture models for real-time tracking. In: Proceedings. 1999 IEEE computer society conference on computer vision and pattern recognition (Cat. No PR00149). vol. 2, pp. 246–252. IEEE (1999)
56. Sultani, W., Chen, C., Shah, M.: Real-world anomaly detection in surveillance videos. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 6479–6488 (2018)
57. Tan, H., Zhai, Y., Liu, Y., Zhang, M.: Fast anomaly detection in traffic surveillance video based on robust sparse optical flow. In: 2016 IEEE international conference on acoustics, speech and signal processing (ICASSP). pp. 1976–1980. IEEE (2016)
58. Tripathi, R.K., Jalal, A.S., Bhatnagar, C.: A framework for abandoned object detection from video surveillance. In: 2013 Fourth national conference on computer vision, pattern recognition, image processing and graphics (NCVPRIPG). pp. 1–4. IEEE (2013)
59. Veni, S., Anand, R., Santosh, B.: Road accident detection and severity determination from cctv surveillance. In: Advances in Distributed Computing and Machine Learning, pp. 247–256. Springer (2021)
60. Wang, C., Musaev, A., Sheinidashtegol, P., Atkison, T.: Towards detection of abnormal vehicle behavior using traffic cameras. In: International Conference on Big Data. pp. 125–136. Springer (2019)
61. Wei, J., Zhao, J., Zhao, Y., Zhao, Z.: Unsupervised anomaly detection for traffic surveillance based on background modeling. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. pp. 129–136 (2018)
62. Wei, L., Hong-ying, D.: Real-time road congestion detection based on image texture analysis. Procedia engineering **137**, 196–201 (2016)
63. Wojke, N., Bewley, A., Paulus, D.: Simple online and realtime tracking with a deep association metric. In: 2017 IEEE international conference on image processing (ICIP). pp. 3645–3649. IEEE (2017)
64. Wren, C.R., Azarbayejani, A., Darrell, T., Pentland, A.P.: Pfinder: Real-time tracking of the human body. IEEE Transactions on pattern analysis and machine intelligence **19**(7), 780–785 (1997)
65. Xia, S., Xiong, J., Liu, Y., Li, G.: Vision-based traffic accident detection using matrix approximation. In: 2015 10th Asian Control Conference (ASCC). pp. 1–5. IEEE (2015)
66. Yun, K., Jeong, H., Yi, K.M., Kim, S.W., Choi, J.Y.: Motion interaction field for accident detection in traffic surveillance video. In: 2014 22nd International Conference on Pattern Recognition. pp. 3062–3067. IEEE (2014)
67. Zhang, X., Feng, Y., Angeloudis, P., Demiris, Y.: Monocular visual traffic surveillance: A review. IEEE Transactions on Intelligent Transportation Systems (2022)
68. Zivkovic, Z.: Improved adaptive gaussian mixture model for background subtraction. In: Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004. vol. 2, pp. 28–31. IEEE (2004)

Authors Index