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 978-3-942952-91-0

www.ibai-publishing.org

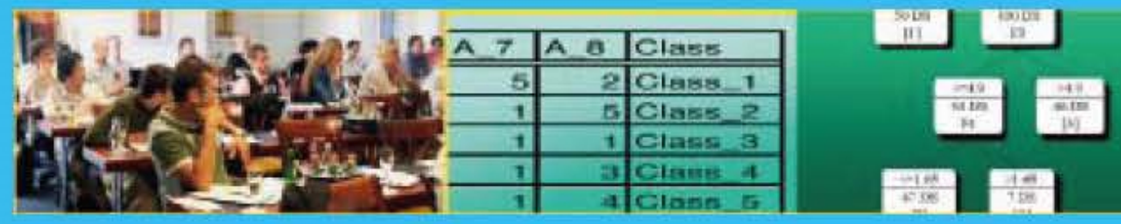
ISBN 978-3-942952-91-0

Advances in Data Mining, Proceedings, ICDM 2022

Petra Perner

Petra Perner (Ed.)

Advances in Data Mining



22nd Industrial Conference on Data Mining, ICDM 2022
New York, USA July 13-17, 2022

Proceedings

ibai - publishing

Petra Perner (Ed.)

Advances in Data Mining

Applications and Theoretical Aspects

22nd Industrial Conference, ICDM 2022
New York, USA, July 13 – 17 2022
Proceedings

ibai Publishing
www.ibai-publishing.org

Volume Editor

Prof. Dr. Petra Perner
Institute of Computer Vision and Applied Computer Sciences IBaI

Hertha-Lindner-Str. 10-12
01067 Dresden, Germany

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 978-3-942952-91-0

Editorial

The twenty-second event of the Industrial Conference on Data Mining ICDM was held in New York again (www.data-mining-forum.de) running under the umbrella of the World Congress on "The Frontiers in Intelligent Data and Signal Analysis, DSA 2022" (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 conference is an application-oriented conference. In recent years, we have seen a decline in interest in application-oriented research. Probably also because now innovative foundations are focused on oriented research, which requires not so time-consuming work with the experts on site. But we hope that interest will stabilize once we have overcome the consequences of Corona.

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 Analysis 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

22nd Industrial Conference on Data Mining ICDM 2022

www.data-mining-forum.de

Chair

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

Program Committee

Table of Content

Key Nodes Discovering in Social Network with Temporal Attributed PageRank

Jinshuo Liu[1], Rodrigue Edo[1][0000−0001−6898−1464], Juan Deng[1], and Jeff Z.Pan[2]

[1] School of National Cybersecurity, Wuhan University, P.R.China
[2] University of Aberdeen, UK

Abstract. The PageRank (PR) is widely used to mine the key nodes in social networks. However, traditional PR divides the nodes' influence to the father nodes incorrectly and evenly in dynamic social networks. To address these issues, we propose a temporal attributed PageRank (taPR) with three attributes: intimacy, interaction, and information content. Extensive experiments on large data-sets, including four Higgs Twitter subsets from the Stanford Network Analysis Project (SNAP) and three preprocessed topic networks (20,841 nodes, 11,361 nodes, and 32,159 nodes) from the Chinese Sina-microblog, have validated the proposed models. taPR outperforms seven cutting-edge algorithms and the baseline PageRank in identifying key nodes. Experiments confirm that attributes such as interaction and intimacy can influence propagation. We achieve GPU parallelization of the proposed model to accelerate the calculation further.

Keywords: Key nodes · Multi-target Evaluation Metric · Node embedding · Temporal attributed PageRank.

1 Introduction

Timely discovering the key nodes of malicious information is just as important as it is to timely discovering the malicious information of the social network [9]. The key nodes, in the social network, may be referred to from two perspectives: nodes have a greater influence on other nodes and bond them together [1]; or nodes are more important than other nodes in information exchange [19]. Research on key node discovery can further benefit other research like link prediction, recommendation systems of online marketing, emergency monitoring, etc [15].

The existing methods of discovering key nodes from the information dissemination process can be divided into three categories: based on network topology, users' social behavior, and users' interaction.

The network topology-based approaches rely heavily on the outcomes of complex networks[14], and adopt metrics like degree, centrality, and PageRank, etc.

There can be many different ways to behave on social media networks. Among these, users can follow each other,get notified of a new message. They can post or leave content, share, reply, and show appreciation for other people's content

(like or dislike). Analyzing these behaviors can reflect the influence among users. The metric can be the importance, popularity, influence, authority, relevance, similarity, the proximity of the nodes, etc.

By studying the interaction between nodes during the information dissemination process of social networks, key nodes can also be obtained. Bakshy et al. [3] construct the tree topology of the information dissemination process, then calculate the influence of each node. Remero et al. [17] analyze the scope and time of the news spread on Twitter and prove that the message with a longer communication time is more influential.

Since each model emphasizes specific attributes, none of the three methods above provides objective evaluation metrics.It is unfair to compare the characteristics of a node from metrics of one method with the metrics of another method

In addition, from the above drawbacks, the traditional PageRank can neither correctly calculate out the influence of node, nor transfer the influence from one node to other nodes. In this paper, thus, we propose a temporal attributed PageRank to solve the problem of wrongly transferring influence, by adding time as one of the attributes to the PageRank. To more accurately calculate the node influence of the social network, we consider more attributes of the node such as intimacy, interaction, and content.

The outline of the paper is as follows. Section 2 introduces some relevant background materials. In section 3, we propose our key nodes discovering method with taPR. In section 4, we devise multiple metrics for evaluating the performance of the model. Section 5 describes the experiments and the discussion. Section 6 is the parallelization part of the method. We summarize the work in section 7.

2 BACKGROUND

2.1 Node Embedding

Node2vec [10] is an algorithm that embeds a graph's nodes into a vector space while keeping their attributes. Node2vec models include the node's network neighborhood and a biased random walk procedure. The challenges of Node2vec relate to the choice of properties, scalability, and dimensionality of the embedding.A good representation of nodes should preserve the structure of the graph and the connections between individual nodes. However, a network with too many criteria and properties may hinder our analysis. On the other hand, the representation should be scalable to process large graphs with millions of nodes. A higher dimension of vectors may lead to more precious construction of the network, which may lead to worse link prediction accuracy and higher time & space complexity. Since the information dissemination network is so huge and complicated, for the request of real-time analysis, the construction of the expression of the nodes need not be too time-consuming.

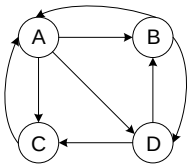


Fig. 1. The example of PageRank with four nodes.

$$\mathbf{M} = \begin{bmatrix} 0 & 1/2 & 1 & 0 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 1/2 & 0 & 0 \end{bmatrix}$$

Fig. 2. The corresponding transfer matrix of Fig. 1.

2.2 PageRank & Temporal PageRank

2.3 PageRank

The page rank system was first introduced in 1998 by Sergey Brin and Larry Page [4]. It's an algorithm used to determine the rank of a web page based on the importance of other pages associated with it. Since its discovery, many researchers have used it to locate key nodes in social networks, leading to significant results. One example can be given by Jianshu Weng [5], who used PageRank to identify influential users of micro-blogging services such as Twitter, Weibo, etc.

$$\pi^{(k+1)} = (1-\sigma)e^T + \sigma\pi^{(k)t}V \tag{1}$$

$\pi^{(k)T}$ and $\pi^{(k+1)T}$ represent the PageRank value before and after the update. σ is the damping coefficient, is the unit vector. V is the voting matrix. The iterative PageRank algorithm can also be improved as:

$$V' = \sigma MV + (1-\sigma)e \tag{2}$$

The time attribute of PageRank has been the subject of some existing research. Taher H. Haveliwala [11] considered the time order query combined with topics and interests to improve the PageRank. Nevertheless, he has not been concerned with the query's time difference Δ, only the order of the query. Fiala [8] associates a temporal attribute to a citation network with a special focus on public citation time and co-authorship. David F. Gleich et al. [13] proposed $v(t)$ as a probability distribution of the places. But they have not defined a precise transfer matrix over time. [2] Haim Avron and Lior Horesh proposed an efficient local algorithm approximating the time-dependent personalized PageRank vector x. Despite considering time, they have not examined actions and their associated effects based on when and how users take actions. Hu Weishu et al. proposed a temporal PageRank based on three factors, including the Built-Up Time-Length Factor (BTF), Frequency Factor (FF), and Similarity Factor (SF). According to them, BTF can influence information dissemination. They've also analyzed the effects of time on FF and SF. Therefore, Rosenshtein Polina & Gionis Aristides [18] introduced Temporal PageRank, which accounts for the possibility that nodes' importance can change over time depending on the distribution of edges.

3 KEY NODES DISCOVERING WITH TEMPORAL PAGERANK

3.1 Problem Setup

Definition (Dynamic Directed Graph, DDG)
Let $G = (N, T)$ be graph G's time-stamped information distribution function. N is the set of nodes $N = n_i|0 \ll i \ll |N|$; T, the set of edges $T = t_j|0 \ll j \ll |T|$. Edges represent the directed path of information between nodes, time-stamped from the start of the diffusion (see Fig.3). The amount of diffused information, correlated with time, user characteristics, interaction characteristics, and intimacy factors, travels in the opposite direction to the diffused influence.

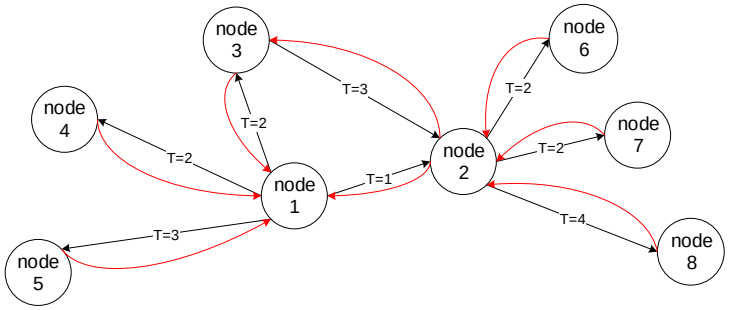


Fig. 3. Dynamic information dissemination directed graph.
Blackline represents the direction of the information disseminated. The red line represents the direction of its influence. The sense of information is opposed to that of influence.

3.2 Temporal Page Rank (taPR)

In this section, we propose the temporal attributed PageRank to estimate the dynamic influence of nodes in the social network. The factors that affect two nodes to relay the information may be related to the social and psychological echo. So we devise the model of calculating the connecting weight based on the topology, the interactions, the intimacies, and the time between nodes. Specially, we consider the effect of time on topology, interaction, and intimacies. As time factors, we considered the time duration (δ) of the dissemination, the frequency, and the connection period. For example, if Node A recently connected to Node B at midnight for a long time, their connection is more important than A's other connections.

taPR model: The iterative formula of the page rank algorithm is improved as taPR. From the updated influence $(k + 1)$ at time T, with the sum of influences of all the previous time kT.

$$V^{(k+1)T} = \sum_{i=1}^{3} A_{kT}^{i}(t) V^{(k)T!} + (1-\sigma)e \tag{3}$$

V is the voting matrix, A_{kt}^{3} is the coefficient matrix. When $i = 1$, A_{kt}^{1} is the time function of the Node2vec. A_{kt}^{2} is organized by the interaction features between the nodes, such as 'retweet' and 'reply' of twitter. A_{kt}^{3} is composed of the intimacy features. σ is the standard of the time of all the considered interactions.

3.3 Realization of the Transferred Matrix $V^{(k)T!}$

To optimize the $V^{(k)T!}$ in equation(3), deducting the computation complexity, we design the methods to realize the transferred matrix.

Because influence transfer operates in the inverse direction of information dissemination, the time-stamped information dissemination graph (black line in Fig. 3) will transform into an influence transfer graph (red line in Fig. 3).

When the transferred influence is combined with the time, the transferred influence should obey the following rules:

1. The generated influence corresponds to the time Δ. The influence should be split by the time Δ.
2. When the node transfers the influence, it should transfer the total influence to the father node after received the father node's time.
3. The sum of the transferred influence equals to the influence itself.
4. Initialize the node's influence as 1, and the corresponding time as the first received time.

In order to avoid the exhausting calculation, considering the second is accurate, the time-stamp can be normalized as:

$$T_{ij} = \lfloor \frac{t_{j \to i} - t_0}{\Delta} \rfloor \tag{4}$$

Δ is the normalized factor. t_{ij} is the exact time-stamp of node j receiving the information from node i. The time-slot T_i can be calculated out by orderly crisscrossing all the timestamps of its normalized inbound and outbound links. M_i is the number of time-slots for node i.

$$T_i = \{m | 0 \leq m \leq M_i\} \tag{5}$$

3.4 Optimization of Matrix $A_{T+1}V_{T+1}$

Since the volume of nodes of the directed graph is huge, to accelerate the computation, we simplify the high dimensional computation of the coefficient matrix. We set A as the partition matrix:

$$A_{T+1} = \begin{pmatrix} A_{T+\delta} & A_2 \\ A_3 & A_4 \end{pmatrix} \tag{6}$$

Compared with time T, the coefficient matrix A at time T+1, the positive δ means the new relationships constructed based on the previous nodes; The negative δ means the

deduct of the relationship based on the previous nodes such as canceling the attention. A_2 and A_3 represent the newly generated nodes that construct the new relationships with old nodes, which may be different since it is a directed graph; A_4 means the totally new relationships within these new nodes. φ means the new force vectors for the new nodes compared with the previous vector V.

$$A_{T+1}V_{T+1} = \begin{pmatrix} A_T + \delta & A_2 \\ A_3 & A_4 \end{pmatrix} \begin{pmatrix} V_T \\ \varphi \end{pmatrix} \tag{7}$$

At time $t+1$, if the newly generated nodes only disseminate the information out, and don't receive the information, the last row of the M should be zeros. The transferring matrix would not be updated until the new nodes start receiving the information. Until then, the transferring matrix has the meaning and has the value for the last row. At time $t+1$, the updated force can be separately calculated as A_3V_T , $A_4\varphi$, δV_T, $A_2\varphi$, and A_TV_T. Since A_3V_T ,$A_4\varphi$, δV_T, and $A_2\varphi$ correlate with new nodes, the dimension of them is the number of the newly generated nodes.

3.5 Optimization of Ranking

Because the target is to extract the top n nodes after ranking, instead of ranking the whole nodes at time $t+1$, we only rank the new nodes with value$A_3V_T + A_4\varphi$, and extract top n nodes from whole nodes. For the old nodes, from the previous time T, distil the top $n+m$ nodes, then calculate $A_TV_T + \delta V_T + A_2\varphi$. Finally, rank the nodes, and extract the top n nodes from the emerged old $n+m$ nodes and n new nodes.

3.6 Algorithm Analysis

Since the basic A_T and V_T can be pre-computed, the time complexity of the Node2vc of one node for a new random walk is $O(1)$. The time complexity of calculating the intimacy factor and interaction feature is also $O(1)$ for one node of the new iteration. The time complexity of V_T is also $O(1)$. So, the total complexity is up to the complexity of ranking. If we choose the heap sort method, then the time complexity is $O(n+m+x)$, where n is the number of top nodes, m is the number of redundancy nodes, and x is the number of new nodes. In a real-world deployment, the overall data is usually too huge. So, one solution is to partition the overall matrix into the partitioned matrices and get the partial solutions. In this way, the huge volume of the directed graph can be divided into the sub-graphs and get the local optima. The other solution is to overlap the matrix with the same nodes set of different times.

Another advantage is that we can avoid accessing the database too frequently. Data analysis demonstrates that our method is not only theoretically effective but also practically meaningful.

The correctness of our method is ensured by dynamic programming and the recurrence relation among the subset of nodes of the direct graph.

4 MULTI- TARGET EVALUATION

Definition 2 (Multi-target Evaluation, ME), the target space is organized by using each target as one dimension. The bigger the volume of ME is, the better the results are. To simplify the calculation, we transform the calculation of the volume of the hyper pyramid into a calculation of the area of the polygon (see Fig. 4).

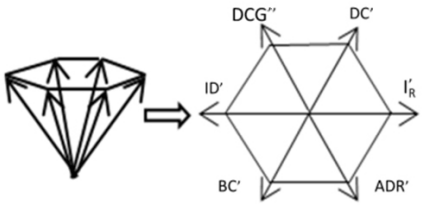


Fig. 4. An example is ME with 6 targets. The left figure is the original calculation of the volume of multi-target. The right figure is the simplification of the area of the corresponding left figure.

There are three types of metrics for finding critical nodes in social media: network topology-related metrics, user-related metrics, and network resilience indicators. Because the aim criteria for various approaches for locating core/bridge/key nodes varies. Also, the nodes from distinct discovery models differ. It's tough to say which method is better and why. As a result, we consider the evaluation model to include these three perspectives.

Within the previous three categories of metrics, the typical metrics comprise Degree Centrality (DC), Closeness Centrality (CC), Eigenvector Centrality (EC), Degree Prestige, In-Degree, intimacy, R, etc. To reduce the complexity of the calculation, we choose the representative set as our dimensions of ME model, the Probability (P), Average Dissemination Range (ADR), DC, EC, In-Degree (ID), BC, and Indicator of Robust (I_R). Thus, the metrics can be not so highly correlated, have good differences, and be as comprehensive as possible. To make the calculation easier, we keep the number of the dimension as even. Then the sum of 6 sub-areas is the final result of the ME (see Fig. 3). We want to emphasize that the data turbulence exists since the data range varies so much in different dimensions. So, we need to normalize the data of each dimension to be within $(0, 1)$.

5 EXPERIMENTS

We extensively evaluate the accuracy of our solution with several types of datasets.

5.1 Implementation and Experiment Setup

We implement a single-threaded prototype method (referenced to as taPR) and PageRank (referred to as PR, a baseline). Both systems are written in Python. We also

provide the parallel version of the prototype in section 6 for performance demonstration. The parameters of taPR and PR are: taPR: When considered individually, each action is granted with 1.0 weight. When mixed, $\omega_{rt(retweet)} = 1.0$, $\omega_{re(replay)} = 0.8$, $\omega_{mt(mention)} = 0.4$, and the default weight for unknown action is set to 0.6. As for the node embeddings, dimension of each outcome is set to 64. The normalization factor Δ is set to 360 minuets (roughly 6 hours) within the dissemination network. $\epsilon = 0.000001$. The maximum iteration is set to 200. $p = q = 1$. Walks per node $r = 10$. Walk length $l = 100$. Content size $k = 10$. The damping factor σ is 0.85 for all our trials. $\alpha = \beta = \gamma = 1$.

- PR: $\epsilon = 0.000001$. The maximum iteration is set to 200. The damping factor is 0.85 for all our trials.
- Super Edge Rank (SER) [12]: $SD_{a_i} = 1, L_{SE} = 11$;
- Hybrid IO-degree (HIO): $\alpha = 0.7$;
- Total Trust Value of relationship dissemination (TTV: $\alpha = 0.7$);

We use two types of data to verify the effectiveness, advantage, and robustness of the methods. One type of data is the Higgs Twitter Dataset of the Standford Network Analysis Project (SNAP). The other type of data is Sina Weibo of China from 2016-2017 crawled by us.

For SNAP, we use the largest Strongly Connected Components (SCC) in trials for each subgraph with only one action type or mixed action type after eliminating all redundant links. The Re-tweet network includes 984 nodes and 3850 edges in its largest strongly connected component. There are 322 nodes and 708 edges in the reply network. There are 1801 nodes and 7069 edges in the mention network. More importantly, the mixed graph includes all three actions, 5548 nodes, and 23378 edges in total (see Table 1). For Sina Weibo, we extract the data of 3 topics to verify our model (see Table 2).

Table 1. DATASET OF **SNAP**

ID	Interaction	Nodes	Edges
1	Retweet (RT)	984	3850
2	Reply(RE)	322	702
3	Mention (MP)	1801	6601
4	Mixed graph (M)	5548	23378

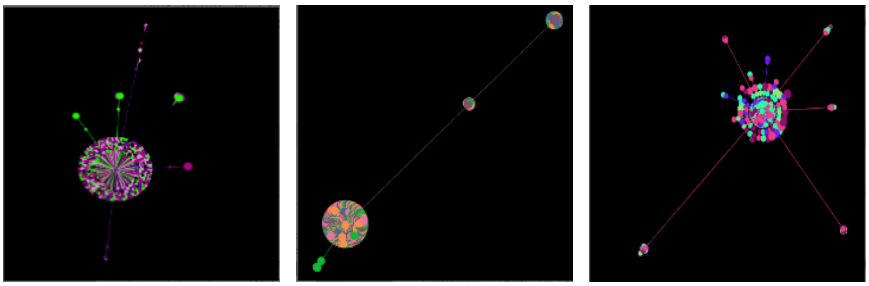
The four types of data of SNAP, and the corresponding interaction, nodes and edges.

We visualize the network of these 3 topics. From Fig. 5a, the Weibo forwarded (FD) mostly are from the original author. The topology is star. But there still exists a few nodes which have more forwarding nodes. In Fig. 5b, most of the forwarded Weibo is from the original Weibo. But during the procedure of forwarding information, there still exists some key nodes with more forwarded information. In Fig. 5c, the forwarded Weibo is more than 30000, and there are many key nodes with many sub-nodes, which leads the information to be more forward.

Table 2. DATASET OF SINA WEIBO

ID	Content	AT	FD	Duration
1	奥斯卡 (Oscar)	高晓松	841	02/28/2017 7:46:00 -03/15/2017 13:07:00
2	一篇回忆 (Memory)	高晓松	1361	03/3/14/2017 8:47:00 -03/142017 20:15:00
3	言不由衷的段落 (None sense)	薛之谦	32159	10/6/2016 18:36:19 -10/16/2016 15:15:32

For Sina Weibo, AT represents the about target. FD represents the 'forward'. Each line of the table is one topic.



(a) sina 1 (b) sina 2 (c) sina 3

Fig. 5. The topology of information dissemination for Sina weibo. Sina 1 means using data set 1 of Sina weibo. Sina 2 and Sina 3 means using the data sets 2 and 3 of the Sina weibo respectively.

5.2 The metrics

Because different methods utilize different metrics to evaluate the effectiveness, we need to find the metrics which can be relatively objective. Since the biggest function of the key nodes' is the higher spreading ability of information, we adopt the Average Dissemination Range of the SIR model as one metric. A good discovering method should have high accuracy under supervised conditions, so we choose P as the second metric. For social media, it is very easy to get the number of the 'forward'. Although the ranking result of 'forward' does not equal to ranking result of the influence, it definitely has the reference significance to validate the effectiveness of ranking influence. So, we adopt Discounted Cumulative Gain (DCG) as the metric together with the 'forward'. Besides the effectiveness, a good method should have good robustness, so we adopt the I_R to evaluate our model. In order to contain more evaluate factors, we devise the ME to comprehensively evaluate the ranking methods (see Table 3).

Table 3. ADOPTED METRICS

metrics	P	SIR	I_R	DCG	ME

The table is the metrics adopted to evaluate the methods for discovering the key nodes.

5.3 Effectiveness of Attributes with SIR

We compare the attributes utilized by our method with other 8 methods, Super Edge Rank (SER), Page Rank (PR), Social Network and Threshold Model (SNTM), Top in-degree (Ti), Top out-degree (To), Hybrid IO-degree (HIO), Top centrality (Tc), Total Trust Value of relationship propagation (TTV).

Table 4 demonstrates that only taPR considers the social relationships, network topology, textual content, intimacy, and time simultaneously. The other 8 methods may neglect some attributes related to affecting information dissemination of social networks.

We define four types of taPR: with intimacy & interaction attributes (taPR); with intimacy & no interaction (NA); no intimacy & with interaction (NI); without intimacy & without interaction (NAI). In order to validate the effectiveness of the attributes of taPR, we select the top 50 nodes from SNAP4 separately using taPR, NA, NI, and NAI. With different sets of 50 nodes, we run the SIR simulation program for 1600 seconds, and then calculate the mean infected nodes of the SIR model. We compare the results of the NA vs NI and NA vs NAI.

The slightly higher infection rate of taPR indicates that both two factors have made joint positive efforts in depicting the network. Figs. 6 and 66 manifest that NI (Interaction factor) is better than NA(intimacy factor). NA is better than NAI on average. More concisely, these figures verify the initial assumptions that our node's multiple factors contribute positively to our final ranking results, and the interaction attribute has the greater impact to the nodes compared with intimacy.

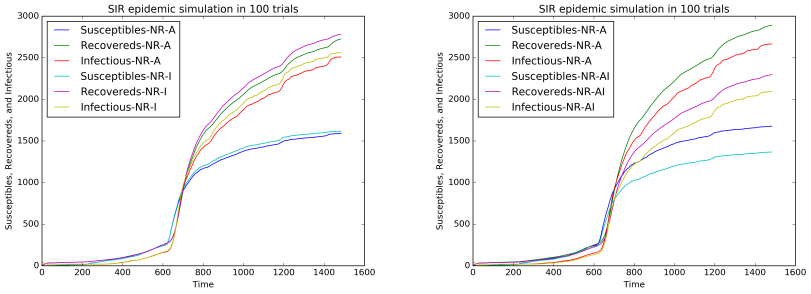
Table 4. THE ATTRIBUTES OF COMMON 9 METHODS

Research	Social relationship	Network topology	Textual content	Intimacy	Time
SER [10]	√	√	√	×	×
SNTM [4]	√	×	√	×	×
To [16]	√	×	√	×	×
Ti [16]	√	×	√	×	×
HIO [16]	√	×	√	×	×
Tc [16]	√	×	√	×	×
TTV [16]	√	×	√	×	×
PR	×	√	×	×	×
taPR	√	√	√	√	√

Rank (PR), Social Network and Threshold Model (SNTM), Top in-degree (Ti), Top out-degree (To), Hybrid IO-degree (HIO), Top centrality (Tc), Total Trust Value of relationship propagation (TTV)

5.4 Average Dissemination Range(ADR) with Suceptible-infectious-removed (SIR)

We firstly adopt the top 50 nodes selected by taPR and PR as the seed nodes separately and then run the SIR simulation program 1600 seconds on the SNAP1, SNAP2, SNAP3, and SNAP4 data set. We use no less than 50 nodes, because we want to make our method be more stable. Also, because the data size of SNAP subsets is not so big, which SNAP2 even only has 322 nodes totally. We calculate the mean value of Susceptible, Infective, and Recoveries (shown as Fig. 3).

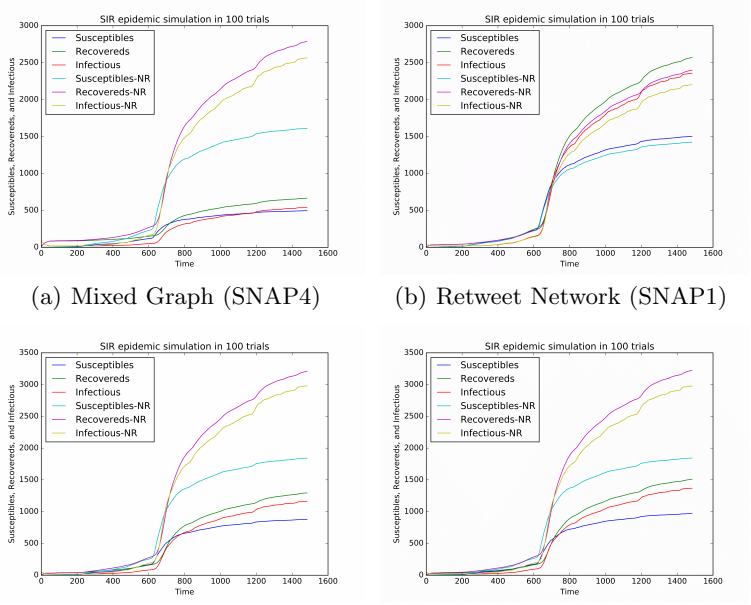


(a) NA vs NI (b) NA vs NAI

Fig. 6. Average SIR Growth Over 100 Trials in One-action Networks. NI is the Interaction factor. NA is the intimacy factor.

Fig. 6 shows the results of NA vs NI and NA vs NAI. The slight higher infection rate of figure 2 in taPR indicates that both two factors have made joint efforts in depicting the network more discriminatingly; Figs 3a and 3b manifest the performance between each factor: NI (Interaction factor) is better than NA(intimacy factor), and NA is better

than NAI on average. More concisely, these figures verify that the initial assumptions about the factors contribute positively to our final ranking results.



(a) Mixed Graph (SNAP4) (b) Retweet Network (SNAP1)

(c) Reply Network (SNAP2) (d) Mention Network (SNAP3)

Fig. 7. Average SIR Growth Over 100 Trials in One-action Networks on dataset SNAP1, SNAP2, SNAP3, and SNAP4.

Figs7. 7a, 7b, 7c, and 7d obviously demonstrate that our selected nodes from taPR eventually infected more population than the traditional PageRank, approximately $5 = \frac{3000}{600}$ times, especially in the action-mixed network (SNAP1). In the re-tweet network (SNAP2), although the results from taPR have not exceeded PR with minor defeat, our approach outperforms the baseline PR in other networks (SNAP3, SNAP4).

Secondly, besides the baseline method PR, we adopt other 7 state-of-art methods to validate the efficiency of taPR (see Table 5). We only compare the infect rate using the SNAP4 data. With the top 50 nodes selected by these methods, after 1600 seconds, taPR gets the biggest number of the infected nodes 2500. TTV has also a relatively bigger range of infected nodes.

Table 5. COMPARISION WITH STATE-OF-ART METHODS.

Algorithm	taPR	PR	SER	Ti	To	HIO	Tc	TTV
SNAP 4	2500	600	1283	1650	1543	1809	1346	2130

The final nodes are infected from 50 nodes after 1600 seconds.

5.5 Tolerance of Noisy & Robust (I_R)

To validate that our method has the advantage of robustness compared with other state-of-art methods, we adopt the I_R.

We design the experiments in two ways: (1) Modify the edges by adding or removing the edges randomly, and then calculate I_1. Since the nodes of Sina1 are only 841, we randomly repeat 10 times changing 50 edges of these three Sina datasets. (2) Forge the 50 fans from the nodes randomly, and then calculate I_2.

Table 10 shows that I_1 and I_2 of taPR are the smallest compared with other methods. So, taPR is the most stable and robust among these methods.

5.6 Evaluation Metric of DCG

We use the DCG_{10} metric to evaluate the effectiveness and advantage of our model taPR over other top 10 data mining techniques.

Table 6. TOP 10 USERS OF SINA1 DATA WITH 8 METHODS.

UserID	**Forward**	taPR	PR	SER	Ti	To	HIO	Tc	TTV
6165993727	**1**	1	1	1	1	1	1	1	1
6039312405	**2**	2	2	2	2	2	2	2	2
3055428041	**5**	3	3	5	3	5	3	3	3
5520426292	**4**	4	4	4	4	4	4	4	4
3674212991	**3**	5	5	3	5	7	5	5	5
3055428041	**6**	6	8	7	8	6	6	8	8
6019822564	**7**	7	10	9	9	9	7	10	10
6023888756	**8**	8	7	10	7	10	9	7	7
5816645929	**9**	9	9	6	6	8	10	9	9
5330228462	**10**	10	12	12	12	13	8	12	12

The content of the table means the ranking number. For example, the third row and second column of the table mean UserID 6039312405 is ranked as 2 by the method of 'Forward num'.

Table 7. DCG''_{10} OF 8 METHODS.

Algorithm	taPR	PR	SER	Ti	To	HIO	Tc	TTV
DCG''_{10}	0	5.276	5.232	4.143	4.146	2.037	3.700	4.262

The table is the result of DCG''_{10}. The basic relevance rel_i is calculated according to the 'forward' of Table 6 using equation 23.

Table 6 shows that the top 5 nodes of taPR, PR, and the other seven techniques are identical. The nodes of these nine approaches are similar from the top 6-10. As a consequence, it reveals that our method's output is generally steady and acceptable. Table 7 shows the $DCG10$" top ten nodes for various techniques. Tables 6 and 7 show

that taPR has a lower DCG_{10}", which is 0, indicating that our ranking results are most comparable to the 'forward' ranking result. Tables 6 and 7 indicate that taPR outperforms other methods in finding critical nodes in information propagation.

5.7 Evaluation Metric of ME

We utilize top 50 nodes as the seed nodes to run 10 times using SIR and calculate out the ME with 8 sub-areas. The results are listed (see Table 8). Table 8 shows that taPR has the highest ME, 0.1304, on data of Sina3 Weibo compared with the other seven state-of-art methods, which validates that our ranking method is relatively better.

Table 8. ME OF 8 METHODS

Algorithm	taPR	PR	SER	Ti	To	HIO	Tc	TTV
Sina3	0.1304	0.0568	0.0770	0.0547	0.0526	0.0811	0.3109	0.0955

The ME is calculated out on data from Sina micro-blog 3.

6 Parallelization and Evaluation

In order to achieve the high analysis throughput, we realize our algorithm on CPU as well as on general-purpose GPU platform on Multithread and GPU Parallel Schema [7].

6.1 Parallel realization

We implement the parallel version of our prototype on a hybrid CPU-GPU machine equipped with Intel Core i5 + 8 G memory and NVIDIA Tesla C2050 GPU (Fermi architecture with 448 GPU cores):

In the multithreading CPU version, we parallelize the procedure of I/O data. We fork the threads for Network Embedding, Intimacy Factor, and Interaction Factor calculation respectively. For input and output data, we evenly divide the data into the data blocks and assign each data block to a thread to control the I/O.

For our discovering methods, we parallelize the computation of the Coefficient Matrix, Intimacy Factor, and Ranking at GPU. For coefficient matrix calculation, we adopt the classical GPU method of implementing and optimizing the matrix multiplication. For the intimacy factors, we distribute the vectors of the Topic features, User's basic features, intimacy index to the threads of a block of the GPU and then do the cosine calculation simultaneously [6]. We divide the nodes into groups and assign them to the threads block for ranking. After finishing the bubble sorting for each group, we select the top k nodes from each n node-set and rank again after merged sub-k nodes.

6.2 I/O with CPUs

For data I/O, the experiments show that the speedup increases as the number of CPU threads increases but decreases once the number of CPU threads exceeds the number

of CPU cores. To reduce cache misses, we should calculate the Network embedding and Interaction Factor as close to the cache as feasible.

The initial I/O speed is not so high because other tasks may take up the CPU. When the thread number is 4, the I/O speed is the highest close to the hard disk's transportation limitation.

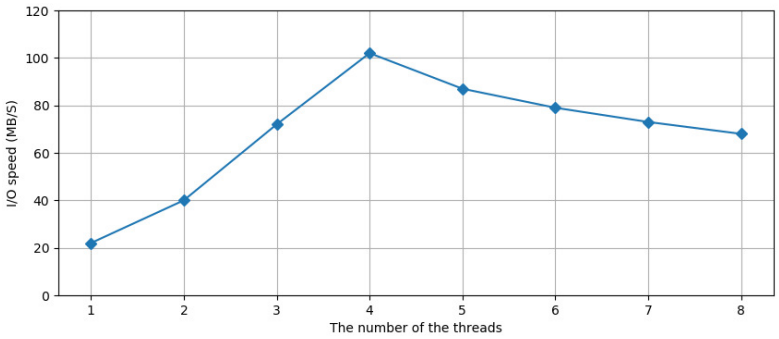


Fig. 8. The speed of I/O with the increases of threads.

6.3 GPU Acceleration

We evaluate the performance of the most time-consuming calculation Coefficient Matrix on GPU with 448 cores distributed across 14 stream multiprocessors and a quad-core CPU. The average speed of the coefficient matrix for the Sina3 Weibo and SNAP4 datasets is 7.8 times. The ranking's average speed is 22.51 times. The average speed of the intimacy factor is 27.36 times.

Why is the speedup lower than expected? It is because of the bottleneck of the frequent data I/O. Although the memory of C2050 is 3G, the intermediate results of a lot of loops will occupy the storage. Another memory limitation is that the computation frequently needs I/O data with external storage. The bandwidth and I/O speed limitation all affect the speedup.

From Table 9, we can see that the speedup of the Sina3 is less than the speedup of the SNAP4 at each step. The reason is that the volume of the data of SNAP is bigger than Sina3. The more the data is, the higher speedup is.

6.4 Total Speedup

We initiate the total speedup with a CPU GPU method using SNAP and the Sina Weibo dataset. When the CPU task is set to 0, all tasks are sent to the GPU, resulting in a total speedup of around 10. Along with the increase in CPU duties, the speedup increases after a modest decrease, showing that adding CPU threads to the GPU can boost performance. As the number of tasks assigned to the GPU diminishes, so does

Table 9. THE GPU SPEEDUP OF THREE STEPS PF TWO DATASETS

	Coefficient matrix	Ranking	Intimacy factor
Sina 3	6.27	6.20	10.30
SNAP 4	9.33	18.82	24.42
Mean speedup	7.80	12.51	17.36

The table shows the speedup with the GPU on two types of data and three steps of the taPR.

its speed, meaning that the main performance is mostly determined by the CPU. The CPU and GPU can interact to maximize computational resources and produce better results.

7 Conclusion

We present a taPR method and evaluation metric for identifying key nodes for disseminating information from a temporal social network. The time effect model on topology, information content, and interaction are all being revised. Our experimental results show that our taPR method can successfully discover key nodes and that the features developed are effective. When compared with other methods, our method is more robust and advantageous. In the future, we intend to investigate the role of a specific small community in information distribution.

Funding: This paper is supported by China Natural Science Founda-tion Natural Science Foundation of China (NSFC) under the Grant No. U193607, National Key R&D Plan of China under the Grant No. 2020YFA0607902.

The corresponding author: J. Deng is with the Computer Science School, Wuhan University, 430071, China. E-mail: dengjuan@whu.edu.cn.

References

1. Aghdam, S.M., Navimipour, N.J.: Opinion leaders selection in the social networks based on trust relationships propagation. Karbala International Journal of Modern Science **2**(2), 88–97 (2016)
2. Avron, H., Horesh, L.: Community detection using time-dependent personalized pagerank. In: International Conference on Machine Learning. pp. 1795–1803. PMLR (2015)
3. Bakshy, E., Hofman, J.M., Mason, W.A., Watts, D.J.: Everyone's an influencer: quantifying influence on twitter. In: Proceedings of the fourth ACM international conference on Web search and data mining. pp. 65–74 (2011)
4. Brin, S., Page, L.: The anatomy of a large-scale hypertextual web search engine. Computer networks and ISDN systems **30**(1-7), 107–117 (1998)
5. Brin, S., Page, L.: The anatomy of a large-scale hypertextual web search engine. Comput. Networks **30**, 107–117 (1998)
6. CUDAmathAPI: Cuda math ap (2021), `https://docs.nvidia.com/cuda/cuda-math-api/modules.html#modules`
7. Dong, Z., Chen, Y., Tricco, T.S., Li, C., Hu, T.: Hunting for vital nodes in complex networks using local information. Scientific Reports **11**(1), 1–13 (2021)

8. Fiala, D.: Time-aware pagerank for bibliographic networks. Journal of Informetrics **6**(3), 370–388 (2012)
9. Gottfried, J., Shearer, E.: News use across social media platforms 2016. pewresearch (2016)
10. Hamilton, W.L., Ying, R., Leskovec, J.: Representation learning on graphs: Methods and applications. arXiv preprint arXiv:1709.05584 (2017)
11. Haveliwala, T.: Topic-sensitive pagerank. In: the 11th international conference on World Wide Web. pp. 517–526. ACM New York, NY (2002)
12. Huang, Y.L., Starbird, K., Orand, M., Stanek, S.A., Pedersen, H.T.: Connected through crisis: Emotional proximity and the spread of misinformation online. In: Proceedings of the 18th ACM conference on computer supported cooperative work & social computing. pp. 969–980 (2015)
13. Kloster, K., Gleich, D.F.: Heat kernel based community detection. In: Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 1386–1395 (2014)
14. Li, Q., Zhou, T., Lü, L., Chen, D.: Identifying influential spreaders by weighted leaderrank. Physica A: Statistical Mechanics and its Applications **404**, 47–55 (2014)
15. Patel, N.G., Rorres, C., Joly, D.O., Brownstein, J.S., Boston, R., Levy, M.Z., Smith, G.: Quantitative methods of identifying the key nodes in the illegal wildlife trade network. Proceedings of the National Academy of Sciences **112**(26), 7948–7953 (2015)
16. Putman, R.D.: Bowling alone: the collapse and revival of american community, ny: Simon and shuster, 2000, 541 p. Política y gobierno pp. 274–276 (2002)
17. Romero, D.M., Galuba, W., Asur, S., Huberman, B.A.: Influence and passivity in social media. In: Joint European Conference on Machine Learning and Knowledge Discovery in Databases. pp. 18–33. Springer (2011)
18. Rozenshtein, P., Gionis, A.: Temporal pagerank. In: ECML/PKDD (2016)
19. White, H.C.: Chains of opportunity. Harvard University Press (2013)

Integrating Physiological Time Series and Clinical Notes with Transformer for Early Prediction of Sepsis

Yuqing Wang[1⋆], Yun Zhao[1⋆], Rachael Callcut[2], and Linda Petzold[1]

[1] Department of Computer Science, University of California, Santa Barbara
[2] UC, Davis Health
wang603@ucsb.edu, yunzhao@cs.ucsb.edu

Abstract. Sepsis is a leading cause of death in the Intensive Care Units (ICU). Early detection of sepsis is critical for patient survival. In this paper, we propose a multimodal Transformer model for early sepsis prediction, using the physiological time series data and clinical notes for each patient within 36 hours of ICU admission. Specifically, we aim to predict sepsis using only the first 12, 18, 24, 30 and 36 hours of laboratory measurements, vital signs, patient demographics, and clinical notes. We evaluate our model on two large critical care datasets: MIMIC-III and eICU-CRD. The proposed method is compared with six baselines. In addition, ablation analysis and case studies are conducted to study the influence of each individual component of the model and the contribution of each data modality for early sepsis prediction. Experimental results demonstrate the effectiveness of our method, which outperforms competitive baselines on all metrics.

Keywords: Deep learning · Transformer · Sepsis.

1 Introduction

Sepsis is a life-threatening organ dysfunction caused by a dysregulated host response to infection [1], contributing to $30\% - 50\%$ of inpatient mortality in the U.S [2]. The capability of early detection of sepsis allows for earlier interventions and treatment, thus improving patient outcomes. Following the widespread adoption of electronic health record (EHR) systems, researchers are particularly interested in using the EHR data to predict sepsis [3–5].

One challenge of using EHR is that it stores both structured data (e.g., vital signs and laboratory measurements) and unstructured data (e.g., physician and nursing notes). Nevertheless, the heterogeneities across modalities increase the difficulty of performing sepsis prediction tasks. Thus, previous research works have focused on analyzing single data modality in isolation [4, 6, 7]. Structured physiological data can represent patients' true physiological signals. However, in the case of sepsis, this data is incomplete and irregular due to urgency in the

⋆ These two authors contributed equally to this paper.

Intensive Care Units. Although unstructured medical notes can help understand patients' conditions more directly by capturing information regarding patients' symptom changes, it is insufficient to use notes alone to determine patients' status without support from physiological data.

To address the issues above, we propose a multimodal Transformer model that incorporates information from both physiological time series data and clinical notes for early prediction of sepsis. We use two large critical care datasets: the Medical Information Mart for Intensive Care III (MIMIC-III) [8] and the eICU Collaborative Research Database (eICU-CRD) [9]. Comprehensive experiments are conducted on the above two datasets to validate our approach, including performance comparison with baselines, ablation analysis, and case studies. Experimental results suggest that our proposed method outperforms six baselines on all metrics on both datasets. In addition, empirical ablation analysis and case studies indicate that each single modality contains unique information that is unavailable to the other modality. Hence, our model improves predictive performance by utilizing both physiological time series data and clinical notes.

The main contributions of this paper are highlighted as follows:

(1) To the best of our knowledge, this is the first Transformer-based model that incorporates multivariate physiological time series data and clinical notes for early sepsis prediction.
(2) Our experimental results indicate that both modalities complement each other. Thus, our method with both physiological data and clinical notes results in the best overall performance compared with unimodal methods. When using both modalities, our method outperforms competitive baselines on all metrics.
(3) We perform attention mechanism visualization on clinical notes to improve the interpretability regarding the patients' status, which is not available in physiological data. In addition, distinctive distributions of physiological features between sepsis and non-sepsis patients demonstrate the unique information contained in physiological data but not in clinical notes.

The remainder of this paper is organized as follows. Section 2 describes related work. The formal problem description is in Section 3. The proposed model is outlined in Section 4. Section 5 describes the datasets we use for evaluation. Experiments and results are discussed in Section 6. Finally, our conclusions are presented in Section 7.

2 Related Work

In this section we review related work on clinical notes and physiological time series modeling, as well as multimodal methods in the clinical domain.

2.1 Clinical Notes Modeling

With the increasing availability of clinical notes over the past several years, there has been notable progress in understanding and using clinical text data to improve

clinical prediction outcomes. Natural language processing (NLP) and information retrieval techniques have been widely applied on different types of clinical tasks, such as clinical relation extraction [10], de-identification of clinical notes [11], and clinical question answering [12]. One common method for text representation is word embedding. In recent years, the appearance of the Transformer-based BERT [13] has offered an advantage over previous word embedding methods such as Word2Vec [14] and GloVe [15] since it produces word representations that are dynamically informed by the words around them, which can effectively capture information from both the left and right contexts. In the clinical domain, BioBERT was pre-trained on PubMed abstracts and articles and was able to better identify biomedical entities and boundaries than base BERT [16]. Base BERT and BioBERT have been further fine-tuned on the MIMIC-III dataset [8] and released as ClinicalBERT and Clinical BioBERT, respectively [17].

2.2 Physiological Time Series Modeling

Previous studies applied classical models such as Gaussian process (GP) and linear dynamical systems (LDS) to clinical time series modeling [18, 19]. Given the growing availability of clinical data, recent studies demonstrate that RNN-based deep learning (DL) methods have become sought-after alternatives in clinical sequence modeling [4, 20]. More recently, an attention-based DL model has been proposed for clinical time series modeling [21].

2.3 Multimodal Methods in the Clinical Domain

Multimodal representation learning is a fundamentally complex problem due to multiple sources of information [22]. Undeniably, multiple sources of data can provide complementary information, enabling more robust predictions [23]. In the clinical domain, predictive models have been developed by integrating continuous monitoring data and discrete clinical event sequences [24]. Combinations of multiple modalities such as clinical texts, procedures, medications, and laboratory measurements have shown improved performance on inpatient mortality, length of stay, and 30-day readmission prediction tasks [25]. Unstructured clinical notes combined with structured measurements have been used for survival analysis of ICU trauma patients [26].

3 Problem Definition

For a patient cohort consisting of P patients, the multivariate physiological time series (MPTS) data associated with each patient can be expressed as $\{\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \cdots, \boldsymbol{x}^{(M)}\} \in \mathbb{R}^{L \times M}$ with $\boldsymbol{x}^{(j)^T} = \{x_1^{(j)}, x_2^{(j)}, \cdots, x_L^{(j)}\}$. M and L represent the number of clinical features and the number of hours after admission, respectively. In addition, for each patient, sequences of clinical notes are used. The true label of each patient's clinical outcome is $y \in \{0, 1\}$ (1 indicates sepsis and 0 indicates non-sepsis). Altogether, each of our datasets can be represented

as $\{(\mathbf{X}_i, \mathbf{C}_i, \mathrm{Y}_i)|i = 1, 2, \cdots, P\}$ where $\mathbf{X}_i, \mathbf{C}_i, \mathrm{Y}_i$ are the respective MPTS sequence, available clinical notes within L hours, and the class label for patient i. We formulate sepsis prediction as a binary classification task, for which the goal is to learn a mapping:

$$(\mathbf{X}_i, \mathbf{C}_i) \rightarrow \mathrm{Prob}(\mathrm{Y}_i = 1|(\mathbf{X}_i, \mathbf{C}_i)),$$

where $i = 1, 2, \cdots, P$. In other words, MPTS data and clinical notes are used simultaneously to predict whether ICU patients admitted through the Emergency Department will develop sepsis.

4 Methods

In this section we propose the multimodal Transformer modeling framework. The model structure is illustrated in Figure 1.

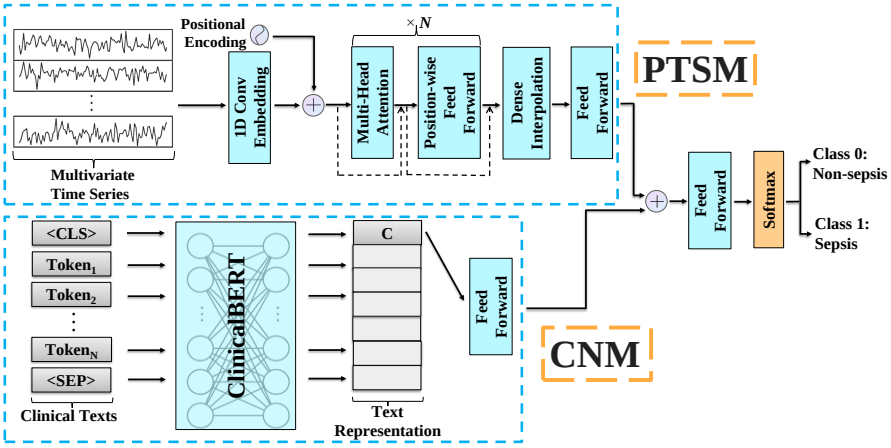


Fig. 1: An overview of the multimodal Transformer framework. The physiological time series model (PTSM) consists primarily of sequence embeddings, a stack of N Transformer encoder layers (multi-head self-attention sublayer and position-wise FNN sublayer) with residual connection around each sublayer, dense interpolation, and FNN. The clinical notes model (CNM) consists of text representations using ClinicalBERT, and the output $\langle$ CLS $\rangle$ representation is then used to feed into FNN. The output representations from PTSM and CNM are concatenated and fed into FNN, and the final Softmax layer is used for the binary classification task.

4.1 Clinical Notes Model (CNM)

The CNM is composed of clinical text representations using ClinicalBERT [17] and a feedforward neural network (FNN). The output ⟨ CLS ⟩ representation following ClinicalBERT is fed into FNN.

Transformer We begin by introducing Transformer's architecture [27], the foundation of Bidirectional Encoder Representations from Transformers (BERT), for which we use for clinical text representations. In Transformer [27], the self-attention mechanism enables the model to capture both short- and long-term dependencies, and different attention heads can learn different aspects of attention patterns. In the self-attention layer, an attention function maps a query $\mathbf{Q}$ and a set of key-value pairs $\{\mathbf{K},\mathbf{V}\}$ to an output $\mathbf{O}$. Specifically, a multi-head self-attention sublayer simultaneously transforms the queries, keys and values into H distinct and learnable linear projections, namely

$$\mathbf{Q}^h = \mathbf{Q}\mathbf{W}_h^Q, \mathbf{K}^h = \mathbf{K}\mathbf{W}_h^K, \mathbf{V}^h = \mathbf{V}\mathbf{W}_h^V,$$

where $\mathbf{Q}^h, \mathbf{K}^h, \mathbf{V}^h$ are the respective query matrices, key matrices, and value matrices of the h-th attention head, with $h = \{1, 2, \cdots, H\}$. Here, $\mathbf{W}_h^Q, \mathbf{W}_h^K \in \mathbb{R}^{d_{\text{model}} \times d_k}$, $\mathbf{W}_h^V \in \mathbb{R}^{d_{\text{model}} \times d_v}$ denote learnable parameter matrices and d_{model} is the text embedding dimension. Next, H attention functions are performed in parallel to produce a sequence of vector outputs:

$$\begin{aligned}\mathbf{O}^h &= \text{Attention}(\mathbf{Q}^h, \mathbf{K}^h, \mathbf{V}^h) \\ &= \text{Softmax}\left(\frac{\mathbf{Q}^h {\mathbf{K}^h}^T}{\sqrt{d_k}}\right)\mathbf{V}^h.\end{aligned}$$

Finally, the outputs $\mathbf{O}^1, \mathbf{O}^2, \cdots, \mathbf{O}^H$ are concatenated and linearly projected again to produce the final representation.

Text Representation with ClinicalBERT BERT is a pre-trained language representation based on the Transformer encoder architecture [13, 27]. BERT and its variants have exhibited outstanding performance in various NLP tasks. In medical contexts, ClinicalBERT develops clinically oriented word representations for clinical NLP tasks. Within ClinicalBERT, each token in clinical notes can be expressed as a sum of corresponding token embeddings, segment embeddings, and position embeddings. When feeding multiple sentences into ClinicalBERT, segment embeddings identify the sequence that a token is associated with and position embeddings of each token are a learned set of parameters corresponding to the token's position in the input sequence [28]. We use pre-trained ClinicalBERT for contextual representations of clinical notes.

4.2 Physiological Time Series Model (PTSM)

Inspired by the model architecture of Transformer [21, 27], the PTSM is composed of sequence embeddings, positional encoding, a stack of N identical Transformer encoder layers, and dense interpolation to incorporate temporal order.

Input Embeddings In most NLP models, input embeddings are commonly used to map relatively low-dimensional vectors to high-dimensional vectors, which facilitate sequence modeling [29]. For the same reason, a time sequence embedding is required to capture the dependencies among different features without considering the temporal information [21]. A 1D convolutional layer is employed to obtain the K-dimensional embeddings ($K > M$) at each time step.

Positional Encoding In order to include the MPTS order information, we apply the same sinusoidal functions for the positional encoding layer as [27] to encode the sequential information and add it to the input embeddings of the sequence.

Transformer Encoder We take advantage of the multi-head self-attention mechanism to capture dependencies of sequences. Similar to [27], we employ 8 parallel attention heads. Following the attention output, a position-wise FNN is applied with two 1D convolutional layers with kernel size 1, and a ReLU activation function in between. A residual connection is employed around each of the two sublayers.

Dense Interpolation A concise representation of the output sequence from the Transformer encoder layer is needed since we do not make predictions at each time step [30]. A dense interpolation algorithm is applied on learned temporal representations for partial temporal order encoding. Given MPTS data, the pseudocode to perform dense interpolation is shown in Algorithm 1. Let $\mathbf{e}_l \in \mathbb{R}^{d_k}$

Algorithm 1: Dense Interpolation

Input: time step l, time sequence length L, input embeddings $\mathbf{e}_l$, interpolation coefficient I.
Output: Dense representation $\mathbf{z}$.
for $l = 1$ ***to*** L **do**
  e $= I \times l \,/ L$;
  for $i = 1$ ***to*** I **do**
    $r = pow(1 - abs(e - i)/I, 2)$;
    $\mathbf{z}_i = \mathbf{z}_i + r \times \mathbf{e}_l$;
  end
end

represent the intermediate representation following the Transformer encoder layers. The size of the interpolated embedding vector is $d_k \times I$, where I is the interpolation coefficient. Algorithm 1 mainly focuses on finding the contribution of $\mathbf{e}_l$ to the position i of the final representation $\mathbf{z}$, denoted by r. At each time step l , we obtain e, the relative position in the final vector representation $\mathbf{z}$, and

r is computed as $r = (1 - \frac{|e-i|}{I})^2$. Finally, $\mathbf{z}$ is obtained by matrix multiplication of r and $\mathbf{e}_l$ when we iterate through the time steps of a sequence.

4.3 Incorporating PTSM and CNM

The output representations from PTSM and CNM are concatenated, and the combined latent representation is fed into FNN. We use a Softmax layer as the final layer for the binary classification problem and the loss function is given by

$$-(y \cdot \log(\hat{y})) + (1 - y) \cdot \log(1 - \hat{y}),$$

where y and $\hat{y}$ are the true and predicted labels, respectively.

5 Datasets

We use the MIMIC-III [8] and eICU-CRD [9] datasets to evaluate our method. MIMIC-III, a publicly available single-center clinical dataset, records $61,532$ ICU stays among $58,976$ hospital admissions, including information on $46,520$ patients from Beth Israel Deaconess Medical Center between 2001 and 2012. The eICU-CRD, a multi-center dataset, consists of health data associated with over $200,000$ admissions to ICUs throughout continental United States between 2014 and 2015. Both datasets contain de-identified data, including patient demographics, vital signs, laboratory measurements, severity of illness, diagnosis, and clinical notes.

5.1 Data Preprocessing Pipelines

This section is divided into structured MPTS data preprocessing and unstructured clinical notes preprocessing, respectively.

MPTS Data Preprocessing For both datasets, patient demographics, vital signs and laboratory measurements are extracted for ICU patients admitted through the Emergency Department. A list of clinically reasonable measurement ranges provided by [31] is used to remove outlier values. In total, we extracted 40 and 38 features from the MIMIC-III and eICU-CRD datasets, respectively. Since data was irregularly sampled, we resample the observation time into hourly bins for each feature. We use the mean value to determine feature values for which there are multiple records within an hour. Missing values are imputed by a combination of forward filling (i.e. using the value of the closest past bin with regard to the missing bin) and then backward filling (i.e. using the value of closest future bin with regard to the missing bin). In addition, we remove patients with hospital admission records of less than 12 hours. We use only the first T hours of MPTS data following patient admission for early sepsis prediction, where $T = 12, 18, 24, 30, 36$. For any patient whose measurement recording hours are less than T, his/her existing last-hour measurements are replicated to T. Otherwise, we truncate his/her measurement hours to T such that the MPTS sequence length for all patients are guaranteed to be the same.

Clinical Notes Preprocessing We use all the available clinical notes between hour 1 and hour 36 after ICU admission. If we use the first T hours of MPTS data, then all the available notes up to T hours are extracted for each patient. Over each interval of T hours, for each patient we concatenate sequences of notes. Next, common text cleaning techniques are applied such as case normalization, stop words removal, and special characters removal are applied to clean the clinical notes. To avoid potential label leakage, we remove sentences containing "sepsis" or "septic". Finally, the processed notes are fed into ClinicalBERT for text representations.

5.2 Sepsis Labeling

We use the Angus criteria [32], which is an International Classification of Diseases, Ninth Revision, Clinical Modification (ICD-9-CM) coding system, to identify sepsis for our datasets. Unlike other sepsis identification methods, it uses the final ICD diagnoses of organ failure and infection rather than feature values from the original datasets, to prevent data leakage issues [4].

5.3 Data Statistics

After data preprocessing, we obtain a population of $18,625$ and $60,593$ ICU admissions for the MIMIC-III and eICU-CRD datasets, respectively. The sizes of positive and negative samples identified by the Angus criteria for each dataset are illustrated in Table 1. Based on the ratio of negative and positive samples, our sepsis prediction task can be considered to be an imbalanced classification problem.

Table 1: Sample sizes of two datasets.

Datasets	**MIMIC-III**	**eICU-CRD**
Total	18,625	60,593
Negative	11,655	55,926
Positive	6,970	4,667

6 Experiments and Results

Our experiments explore: (1) the predictive performance of the multimodal Transformer model on the MIMIC-III and eICU-CRD datasets, (2) the relative importance of individual components of the model through ablation analysis, and (3) case studies on both clinical notes and MPTS data.

6.1 Settings

Both datasets are randomly split, with the training set and testing set of sizes 80% and 20%, respectively. We set aside 20% of the training set for the validation set. Experiments are conducted using the first 12, 18, 24, 30 and 36 hours of

patient demographics, vital signs, laboratory measurements, and clinical notes for the Emergency Department patients on both datasets. All experiments were implemented in Pytorch [33] on one NVIDIA Tesla P100 GPU. We minimize the cross entropy loss with the Adam [34] optimizer for training. The hyperparameter search space for each dataset is listed in Table 2, where the hyperparameter values in bold indicate the optimal values found for our model using both modalities. Note that the batch size and the sequence length choice is limited by the available GPU memory. We perform grid search for hyperparameter optimization.

Table 2: Hyperparameter search space of our model on both datasets. Bold values are the optimal values found using both modalities.

Hyperparameters	**MIMIC-III**	**eICU-CRD**
learning rate	[1e-4,**2e-5**,3e-5,5e-5]	[**1e-5**,2e-5,3e-5,5e-5]
dropout rate	[**0.1**,0.2,0.5]	[**0.1**,0.2,0.5]
batch size	[4,**8**,12]	[4,8,**12**]
activation function	[**ReLU**,SELU,GELU]	[ReLU,SELU,**GELU**]
training epochs	[3,**4**,5]	[3,4,**5**]
sequence length	[256,**512**]	[256,**512**]
♯ of encoder layers N	[3,4,5,**6**]	[3,4,5,**6**]
interpolation coefficient I	[12,**24**,32]	[12,24,**32**]
input embedding dim K	[64,**128**]	[64,**128**]
class weight	[0.5, **0.55**, 0.6, 0.65]	[0.0001,**0.0005**,0.001]

6.2 Baselines and Evaluation Metrics

We compare the performance of our model with the following six baselines where the first component (i.e., LSTM, BiLSTM, GRU) is commonly used for time series modeling and the second component (i.e., Word2Vec, FastText, ELMo) is commonly used for text representations in existing literature. Two components are integrated that support two modalities (i.e., time series and clinical notes).

- **LSTM + CNM** [35]
- **BiLSTM + CNM** [36]
- **GRU + CNM** [37]
- **PTSM + Word2Vec** [14]
- **PTSM + FastText** [38]
- **PTSM + ELMo** [39]

We evaluate our model performance in terms of area under the receiver operating characteristic curve (AUROC), F1 score, recall and precision, which are common metrics for imbalanced classification. In addition to the hyperparameters listed in Table 2, we fine-tune additional hyperparameters for LSTM (number of layers, hidden units), GRU (number of layers, hidden units), Word2Vec (window size, number of negative samples), FastText (maximum length of word n-gram, number of buckets), and ELMo (bidirectional and number of negative samples) as shown in Table 3. The hyperparameter values in bold indicate the optimal values found for our baseline models using both modalities. The number of negative samples is based on the negative sampling algorithm.

Table 3: Hyperparameter search space of baselines on both datasets. Bold values are the optimal values found using both modalities.

	Hyperparameters	**MIMIC-III**	**eICU-CRD**
LSTM	♯ of layers	[1,2,**3**,4]	[1,2,3,**4**]
	hidden units	[100,150,**200**]	[100,150,**200**]
	bidirectional	[**Yes**, No]	[**Yes**, No]
GRU	♯ of layers	[1,2,**3**,4]	[1,2,**3**,4]
	hidden units	[100,150,**200**]	[100,150,**200**]
Word2Vec	window size	[5,10,**20**]	[5,**10**,20]
	♯ of negative samples	[**10**,15,20]	[10,**15**,20]
FastText	max length of word n-gram	[2, **5**, 10]	[**2**, 5, 10]
	♯ of buckets	[**1000**,2000,3000]	[1000,**2000**,3000]
ELMo	bidirectional	[**Yes**, No]	[**Yes**, No]
	♯ of negative samples	[10, 15, **20**, 30]	[10, 15, 20, **30**]

6.3 Results

The results of our method and all baselines using both modalities on two datasets are shown in Table 4a and 4b, respectively. We can see that our method outperforms all baselines on both datasets on all metrics regardless of hours we use. Compared to LSTM and GRU, PTSM benefited from its self-attention mechanism. Specifically, PTSM has direct access to all of the available data in parallel, which leaves no room for information loss. Furthermore, compared to Word2Vec and FastText, CNM (ClinicalBERT) provides dynamic contextualized word representations instead of static embeddings, which brings about flexible text representations. For ELMo, since it is based on BiLSTM, it may not be able to deal with long-term dependencies as well as Transformer-based CNM. In general, all models performed better when supplied with available MPTS data and clinical notes covering more hours.

6.4 Ablation Analysis

To further study the influence of each individual component of our proposed method, we conduct ablation experiments to investigate the influence of individual model components with different data inputs. The results of ablation analysis on both datasets are presented in Table 5a and 5b, respectively. First, we consider the performance of applying MPTS data on PTSM only and clinical notes on CNM only. As can be seen from Table 5, in general, the model with input of solely MPTS data has better performance than that of solely clinical notes. Next, we utilize both data modalities with only hour 1 MPTS data (admission measurements) and available clinical notes within T hours since admission where $T = 12, 18, 24, 30, 36$. When using both modalities, they can bring about comparable results with those of using MPTS data only. Finally, in terms of our full model using full MPTS data and clinical notes, the performance improves with the available data covering

more hours by a margin of 4.3% – 8.5% on AUROC, 4.1% – 7.3% on F1 score, 3.3% – 9.1% on precision, and 3.0% – 7.9% on recall compared with the "best" model performance when using partial data. The ablation analysis suggests that both MPTS data and clinical notes complement and benefit each other and thus the model with both modalities has better performance than the model with single modality.

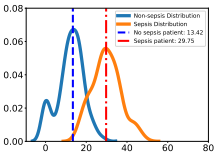
Table 4: Performance comparison for the MIMIC-III and eICU-CRD datasets between the proposed method and six baselines. Hours represent all the data available including MPTS and clinical notes after admission. Experiments are conducted 5 times with different random seeds. The results are shown in the format of mean and standard deviation.

	Hours	12	18	24	30	36
Baseline 1: LSTM + CNM	AUROC	0.854 ± 0.009	0.867 ± 0.008	0.875 ± 0.008	0.878 ± 0.009	0.884 ± 0.008
	F1 Score	0.846 ± 0.006	0.852 ± 0.007	0.856 ± 0.007	0.857 ± 0.006	0.861 ± 0.006
	Precision	0.797 ± 0.006	0.799 ± 0.007	0.801 ± 0.007	0.802 ± 0.006	0.807 ± 0.006
	Recall	0.901 ± 0.006	0.913 ± 0.007	0.918 ± 0.007	0.921 ± 0.006	0.923 ± 0.006
Baseline 2: BiLSTM + CNM	AUROC	0.861 ± 0.008	0.869 ± 0.008	0.878 ± 0.009	0.886 ± 0.009	0.890 ± 0.008
	F1 Score	0.853 ± 0.009	0.858 ± 0.007	0.862 ± 0.008	0.865 ± 0.009	0.869 ± 0.008
	Precision	0.803 ± 0.009	0.808 ± 0.007	0.811 ± 0.008	0.813 ± 0.009	0.816 ± 0.008
	Recall	0.909 ± 0.009	0.914 ± 0.007	0.920 ± 0.008	0.924 ± 0.009	0.930 ± 0.008
Baseline 3: GRU + CNM	AUROC	0.849 ± 0.011	0.856 ± 0.012	0.864 ± 0.011	0.871 ± 0.012	0.876 ± 0.010
	F1 Score	0.842 ± 0.009	0.844 ± 0.010	0.848 ± 0.009	0.851 ± 0.011	0.853 ± 0.012
	Precision	0.795 ± 0.009	0.797 ± 0.010	0.802 ± 0.009	0.805 ± 0.011	0.806 ± 0.012
	Recall	0.896 ± 0.009	0.898 ± 0.010	0.900 ± 0.009	0.903 ± 0.011	0.906 ± 0.012
Baseline 4: PTSM + Word2Vec	AUROC	0.838 ± 0.008	0.851 ± 0.007	0.859 ± 0.007	0.863 ± 0.007	0.872 ± 0.008
	F1 Score	0.830 ± 0.009	0.836 ± 0.008	0.848 ± 0.007	0.851 ± 0.008	0.855 ± 0.009
	Precision	0.792 ± 0.009	0.794 ± 0.008	0.798 ± 0.007	0.800 ± 0.008	0.801 ± 0.009
	Recall	0.872 ± 0.009	0.882 ± 0.008	0.905 ± 0.007	0.910 ± 0.008	0.916 ± 0.009
Baseline 5: PTSM + FastText	AUROC	0.859 ± 0.007	0.868 ± 0.009	0.875 ± 0.012	0.883 ± 0.010	0.889 ± 0.009
	F1 Score	0.851 ± 0.008	0.854 ± 0.009	0.859 ± 0.007	0.861 ± 0.007	0.865 ± 0.009
	Precision	0.801 ± 0.008	0.804 ± 0.009	0.809 ± 0.007	0.811 ± 0.007	0.815 ± 0.009
	Recall	0.907 ± 0.008	0.911 ± 0.009	0.915 ± 0.007	0.918 ± 0.007	0.922 ± 0.009
Baseline 6: PTSM + ELMo	AUROC	0.863 ± 0.005	0.871 ± 0.007	0.880 ± 0.006	0.889 ± 0.007	0.892 ± 0.006
	F1 Score	0.854 ± 0.006	0.859 ± 0.007	0.863 ± 0.006	0.867 ± 0.008	0.870 ± 0.007
	Precision	0.805 ± 0.006	0.810 ± 0.007	0.814 ± 0.006	0.817 ± 0.008	0.819 ± 0.007
	Recall	0.910 ± 0.006	0.915 ± 0.007	0.918 ± 0.006	0.923 ± 0.008	0.928 ± 0.007
Ours: PTSM + CNM	AUROC	**0.902 ± 0.004**	**0.910 ± 0.005**	**0.917 ± 0.005**	**0.923 ± 0.004**	**0.928 ± 0.004**
	F1 Score	**0.881 ± 0.005**	**0.887 ± 0.006**	**0.894 ± 0.004**	**0.907 ± 0.005**	**0.910 ± 0.004**
	Precision	**0.839 ± 0.005**	**0.845 ± 0.006**	**0.852 ± 0.004**	**0.866 ± 0.005**	**0.869 ± 0.004**
	Recall	**0.928 ± 0.005**	**0.933 ± 0.006**	**0.940 ± 0.004**	**0.951 ± 0.005**	**0.955 ± 0.004**

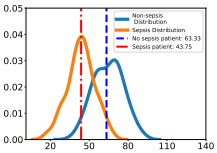
(a) Comparison results on the MIMIC-III testing set.

	Hours	12	18	24	30	36
Baseline 1: LSTM + CNM	AUROC	0.796 ± 0.012	0.801 ± 0.010	0.816 ± 0.009	0.827 ± 0.010	0.830 ± 0.011
	F1 Score	0.787 ± 0.009	0.792 ± 0.008	0.794 ± 0.008	0.796 ± 0.009	0.798 ± 0.010
	Precision	0.773 ± 0.009	0.779 ± 0.008	0.782 ± 0.008	0.783 ± 0.009	0.786 ± 0.010
	Recall	0.802 ± 0.009	0.805 ± 0.008	0.806 ± 0.008	0.809 ± 0.009	0.810 ± 0.010
Baseline 2: BiLSTM + CNM	AUROC	0.802 ± 0.012	0.809 ± 0.011	0.825 ± 0.009	0.833 ± 0.009	0.851 ± 0.009
	F1 Score	0.790 ± 0.008	0.801 ± 0.009	0.813 ± 0.009	0.820 ± 0.008	0.827 ± 0.008
	Precision	0.778 ± 0.008	0.781 ± 0.009	0.785 ± 0.009	0.787 ± 0.008	0.794 ± 0.008
	Recall	0.802 ± 0.008	0.821 ± 0.009	0.844 ± 0.009	0.855 ± 0.008	0.863 ± 0.008
Baseline 3: GRU + CNM	AUROC	0.791 ± 0.007	0.800 ± 0.008	0.813 ± 0.008	0.824 ± 0.009	0.829 ± 0.008
	F1 Score	0.771 ± 0.008	0.782 ± 0.008	0.796 ± 0.007	0.793 ± 0.006	0.797 ± 0.007
	Precision	0.776 ± 0.008	0.780 ± 0.008	0.783 ± 0.007	0.787 ± 0.006	0.792 ± 0.007
	Recall	0.770 ± 0.008	0.784 ± 0.008	0.789 ± 0.007	0.799 ± 0.006	0.803 ± 0.007
Baseline 4: PTSM + Word2Vec	AUROC	0.787 ± 0.009	0.796 ± 0.009	0.811 ± 0.008	0.824 ± 0.010	0.834 ± 0.009
	F1 Score	0.784 ± 0.008	0.788 ± 0.007	0.804 ± 0.007	0.813 ± 0.008	0.821 ± 0.008
	Precision	0.778 ± 0.008	0.780 ± 0.007	0.784 ± 0.007	0.787 ± 0.008	0.792 ± 0.008
	Recall	0.791 ± 0.008	0.796 ± 0.007	0.824 ± 0.007	0.841 ± 0.008	0.852 ± 0.008
Baseline 5: PTSM + FastText	AUROC	0.814 ± 0.012	0.826 ± 0.011	0.838 ± 0.011	0.846 ± 0.010	0.852 ± 0.011
	F1 Score	0.807 ± 0.009	0.815 ± 0.010	0.822 ± 0.010	0.828 ± 0.009	0.834 ± 0.010
	Precision	0.802 ± 0.009	0.805 ± 0.010	0.808 ± 0.010	0.810 ± 0.009	0.813 ± 0.010
	Recall	0.813 ± 0.009	0.826 ± 0.010	0.837 ± 0.010	0.848 ± 0.009	0.856 ± 0.010
Baseline 6: PTSM + ELMo	AUROC	0.812 ± 0.008	0.821 ± 0.009	0.832 ± 0.007	0.844 ± 0.008	0.849 ± 0.009
	F1 Score	0.808 ± 0.007	0.815 ± 0.006	0.819 ± 0.007	0.824 ± 0.008	0.830 ± 0.007
	Precision	0.803 ± 0.007	0.807 ± 0.006	0.809 ± 0.007	0.811 ± 0.008	0.812 ± 0.007
	Recall	0.814 ± 0.007	0.823 ± 0.006	0.829 ± 0.007	0.837 ± 0.008	0.849 ± 0.007
Ours: PTSM + CNM	AUROC	**0.845 ± 0.006**	**0.852 ± 0.005**	**0.861 ± 0.005**	**0.873 ± 0.006**	**0.882 ± 0.004**
	F1 Score	**0.833 ± 0.005**	**0.840 ± 0.004**	**0.845 ± 0.004**	**0.851 ± 0.004**	**0.857 ± 0.003**
	Precision	**0.802 ± 0.005**	**0.807 ± 0.004**	**0.809 ± 0.004**	**0.814 ± 0.004**	**0.818 ± 0.003**
	Recall	**0.866 ± 0.005**	**0.875 ± 0.004**	**0.884 ± 0.004**	**0.892 ± 0.004**	**0.900 ± 0.003**

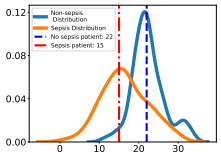
(b) Comparison results on the eICU-CRD testing set.



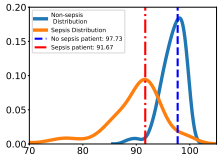
(a) Respiratory Rates



(b) Diastolic Blood Pressure



(c) Bicarbonate Levels



(d) Oxygen Saturation

Fig. 2: Density plots of features. The blue and orange curves are density curves of corresponding features. The blue curve represents no sepsis, and the orange represents sepsis. The dashed vertical lines shows the two patients' feature values.

6.5 Case Studies

We perform case studies to evaluate the uniqueness of each modality in which they may contain information that is inaccessible by the other modality. Figure 3 depicts four self-attention mechanisms in our model which help to understand

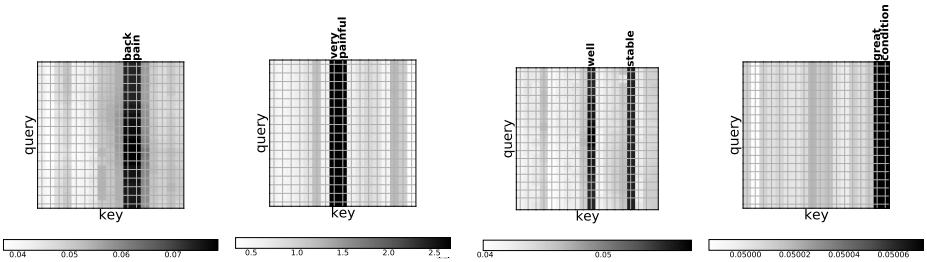
Table 5: Ablation analysis on the influence of different components in our model for the MIMIC-III and eICU-CRD datasets. Experiments are conducted 5 times with different random seeds. The results are shown in the format of mean and standard deviation. Note that hour 1 MPTS indicates that only initial measurements are considered as input instead of a series of measurements. Also, the case that hour 1 clinical notes (i.e. admission notes) with increasing available MPTS data is not considered since the available notes for each patient at the initial time is limited.

	Hours	12	18	24	30	36
MPTS on PTSM only	AUROC	0.827 ± 0.009	0.835 ± 0.008	0.839 ± 0.010	0.842 ± 0.007	0.846 ± 0.008
	F1 Score	0.822 ± 0.006	0.830 ± 0.007	0.831 ± 0.006	0.837 ± 0.008	0.838 ± 0.007
	Precision	0.777 ± 0.006	0.784 ± 0.007	0.771 ± 0.007	0.793 ± 0.008	0.778 ± 0.007
	Recall	0.872 ± 0.006	0.882 ± 0.007	0.900 ± 0.006	0.887 ± 0.008	0.907 ± 0.007
Clinical Notes on CNM only	AUROC	0.790 ± 0.008	0.797 ± 0.007	0.806 ± 0.008	0.812 ± 0.009	0.831 ± 0.007
	F1 Score	0.776 ± 0.007	0.789 ± 0.007	0.799 ± 0.009	0.804 ± 0.008	0.823 ± 0.007
	Precision	0.749 ± 0.008	0.740 ± 0.007	0.784 ± 0.009	0.782 ± 0.009	0.792 ± 0.007
	Recall	0.806 ± 0.007	0.846 ± 0.007	0.814 ± 0.009	0.828 ± 0.008	0.856 ± 0.007
Hour 1 MPTS on PTSM + CNM	AUROC	0.836 ± 0.011	0.839 ± 0.010	0.846 ± 0.008	0.862 ± 0.009	0.871 ± 0.009
	F1 Score	0.830 ± 0.009	0.831 ± 0.008	0.833 ± 0.008	0.847 ± 0.009	0.863 ± 0.010
	Precision	0.795 ± 0.009	0.787 ± 0.008	0.789 ± 0.008	0.794 ± 0.009	0.808 ± 0.010
	Recall	0.869 ± 0.009	0.880 ± 0.008	0.883 ± 0.008	0.907 ± 0.009	0.927 ± 0.010
Ours: PTSM + CNM	AUROC	**0.902 ± 0.004**	**0.910 ± 0.005**	**0.917 ± 0.005**	**0.923 ± 0.004**	**0.928 ± 0.004**
	F1 Score	**0.881 ± 0.005**	**0.887 ± 0.006**	**0.894 ± 0.004**	**0.907 ± 0.005**	**0.910 ± 0.004**
	Precision	**0.839 ± 0.005**	**0.845 ± 0.006**	**0.852 ± 0.004**	**0.866 ± 0.005**	**0.869 ± 0.004**
	Recall	**0.928 ± 0.005**	**0.933 ± 0.006**	**0.940 ± 0.004**	**0.951 ± 0.005**	**0.955 ± 0.004**

(a) Ablation analysis results on the MIMIC-III testing set.

	Hours	12	18	24	30	36
MPTS on PTSM only	AUROC	0.782 ± 0.006	0.788 ± 0.007	0.793 ± 0.008	0.796 ± 0.009	0.817 ± 0.008
	F1 Score	0.773 ± 0.005	0.776 ± 0.006	0.780 ± 0.006	0.781 ± 0.007	0.799 ± 0.006
	Precision	0.706 ± 0.005	0.750 ± 0.006	0.733 ± 0.007	0.727 ± 0.007	0.757 ± 0.006
	Recall	0.780 ± 0.005	0.803 ± 0.006	0.833 ± 0.006	0.844 ± 0.007	0.847 ± 0.006
Clinical Notes on CNM only	AUROC	0.724 ± 0.008	0.733 ± 0.010	0.748 ± 0.009	0.756 ± 0.007	0.778 ± 0.008
	F1 Score	0.717 ± 0.007	0.721 ± 0.006	0.736 ± 0.006	0.742 ± 0.007	0.761 ± 0.008
	Precision	0.695 ± 0.007	0.692 ± 0.007	0.704 ± 0.006	0.708 ± 0.007	0.719 ± 0.008
	Recall	0.740 ± 0.007	0.752 ± 0.006	0.771 ± 0.006	0.780 ± 0.007	0.808 ± 0.007
Hour 1 MPTS on PTSM + CNM	AUROC	0.794 ± 0.011	0.801 ± 0.009	0.814 ± 0.008	0.831 ± 0.009	0.846 ± 0.008
	F1 Score	0.787 ± 0.008	0.792 ± 0.007	0.805 ± 0.007	0.817 ± 0.008	0.823 ± 0.008
	Precision	0.765 ± 0.008	0.768 ± 0.007	0.777 ± 0.007	0.786 ± 0.008	0.792 ± 0.008
	Recall	0.811 ± 0.008	0.818 ± 0.007	0.836 ± 0.007	0.851 ± 0.008	0.857 ± 0.008
Ours: PTSM + CNM	AUROC	**0.845 ± 0.006**	**0.852 ± 0.005**	**0.861 ± 0.005**	**0.873 ± 0.006**	**0.882 ± 0.004**
	F1 Score	**0.833 ± 0.005**	**0.840 ± 0.004**	**0.845 ± 0.004**	**0.851 ± 0.004**	**0.857 ± 0.003**
	Precision	**0.802 ± 0.005**	**0.807 ± 0.004**	**0.809 ± 0.004**	**0.814 ± 0.004**	**0.818 ± 0.003**
	Recall	**0.866 ± 0.005**	**0.875 ± 0.004**	**0.884 ± 0.004**	**0.892 ± 0.004**	**0.900 ± 0.003**

(b) Ablation analysis results on the eICU-CRD testing set.



(a) patient with sepsis (b) patient with sepsis (c) patient without sepsis (d) patient without sepsis

Fig. 3: ClinicalBERT attention mechanism visualization. The x-axis are the query tokens and the y-axis are the key tokens. Panels (a) and (b) are two head attention mechanisms for a patient. The input notes read "remain intubated and feel periodically very painful with back pain while awake during mechanical ventilation". Panels (a) and (b) extract "back pain" and "very painful" as prominent patterns from the two heads, respectively, which provides insight on the patient's critically ill condition. Similarly, panels (c) and (d) are two head attention mechanisms for a patient without sepsis. The input notes include "feel comfortable and tolerating cpap well and vital signs keep stable overall great condition". "Well" and "stable" stand out in panel (c) and "great condition" emerges in panel (d). All of those words are strong indications that the patient is in a relatively benign condition.

patterns in the clinical notes. In all of the panels, the x-axis represents the query tokens and the y-axis represents the key tokens. In panels (a) and (b), we analyze the medical note "remain intubated and feel periodically very painful with back pain while awake during mechanical ventilation" from a patient with sepsis. Panels (a) and (b) are two different head attention mechanisms. Panel (a) indicates "back pain" and panel (b) extracts "very painful" as prominent patterns, respectively. Similarly, panels (c) and (d) are two head attention mechanisms for a patient that ends up with no sepsis. The input note is "feel comfortable and tolerating cpap well and vital signs keep stable overall great condition". CNM finds "well", "stable" and "great condition" in panels (c) and (d), respectively. Both "very painful" and "great condition" help in understanding the patients' conditions and strongly correlate with the final sepsis outcomes. The indications from extracted patterns to patient outcomes show the effectiveness of the ClinicalBERT representations for clinical notes.

Then, we compare some physiological feature values from MPTS data in Figure 2, which plots the univariate distributions of selected features for sepsis and non-sepsis patients, respectively. The orange and blue curves are density curves of observed components of features. The orange curves represent the density curves of sepsis, and the blue ones represent no sepsis. Dashed vertical lines are two patients' corresponding measurement values, who were correctly classified by the proposed model (PTSM + CNM) while misclassified by CNM only. Case studies suggest that single modality does not contain all the possible information that benefits the final prediction. Consequently, using both MPTS data and clinical notes can help obtain more information, which is conducive to the better predictive performance of the model.

7 Conclusion

In this paper, we incorporate multivariate physiological time series data and clinical notes with Transformer for early prediction of sepsis. Comprehensive experiments are conducted on two large critical care datasets, including baseline comparison, ablation analysis, and case studies. Our results demonstrate the effectiveness of our method when using both data modalities, which consistently outperforms competitive baselines on all metrics. Further analysis, and specifically to include clinicians' treatment measures in the input data, are worth exploring.

8 Acknowledgments

This work was funded by the National Institutes for Health (NIH) grant NIH 7R01HL149670.

References

1. Singer, M., Deutschman, C.S., Seymour, C.W., et al.: The third international consensus definitions for sepsis and septic shock (Sepsis-3). Jama, vol. 315, pp. 801-810 (2016).

2. Liu, V., Escobar, G.J., Greene, J.D., et al.: Hospital deaths in patients with sepsis from 2 independent cohorts. Jama, vol. 312, pp. 90-92 (2014).
3. Desautels, T., Calvert, J., Hoffman, J., et al.: Prediction of sepsis in the intensive care unit with minimal electronic health record data: a machine learning approach. JMIR medical informatics, vol. 4, pp. e5909 (2016).
4. Saqib, M., Sha, Y., Wang, M. D.: Early prediction of sepsis in EMR records using traditional ML techniques and deep learning LSTM networks. In: 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), pp. 4038-4041 (2018).
5. Masino, A. J., Harris, M. C., Forsyth, D., et al.: Machine learning models for early sepsis recognition in the neonatal intensive care unit using readily available electronic health record data. PloS one, vol. 14, pp. e0212665 (2019).
6. Lipton, Z. C., Kale, D., Wetzel, R.: Directly modeling missing data in sequences with rnns: Improved classification of clinical time series. In: Machine learning for healthcare conference, pp. 253-270 (2016).
7. Feng, J., Shaib, C., Rudzicz, F.: Explainable clinical decision support from text. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1478-1489 (2020).
8. Johnson, A. E., Pollard, T. J., Shen, L., et al.: MIMIC-III, a freely accessible critical care database. Scientific data, vol. 3, pp. 1-9 (2016).
9. Pollard, T. J., Johnson, A. E., Raffa, J. D., et al.: The eICU Collaborative Research Database, a freely available multi-center database for critical care research. Scientific data, vol. 5, pp. 1-13 (2018).
10. Patrick, J., Li, M.: High accuracy information extraction of medication information from clinical notes: 2009 i2b2 medication extraction challenge. Journal of the American Medical Informatics Association, vol. 17, pp. 524-527 (2010).
11. Deleger, L., Molnar, K., Savova, G., et al.: Large-scale evaluation of automated clinical note de-identification and its impact on information extraction. Journal of the American Medical Informatics Association, vol. 20, pp. 84-94 (2013).
12. Goodwin, T. R., Harabagiu, S. M.: Medical question answering for clinical decision support. In: Proceedings of the 25th ACM international on conference on information and knowledge management, pp. 297-306 (2016).
13. Devlin, J., Chang, M. W., Lee, K., et al.: Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018).
14. Mikolov, T., Sutskever, I., Chen, K., et al.: Distributed representations of words and phrases and their compositionality. In: Advances in neural information processing systems, pp. 3111-3119 (2013).
15. Pennington, J., Socher, R., Manning, C. D.: Glove: Global vectors for word representation. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), pp. 1532-1543 (2014).
16. Lee, J., Yoon, W., Kim, S., et al.: BioBERT: a pre-trained biomedical language representation model for biomedical text mining. Bioinformatics, vol. 36, pp. 1234-1240 (2020).
17. Alsentzer, E., Murphy, J. R., Boag, W., et al.: Publicly available clinical BERT embeddings. arXiv preprint arXiv:1904.03323 (2019).
18. Liu, Z., Wu, L., Hauskrecht, M.: Modeling clinical time series using gaussian process sequences. In: Proceedings of the 2013 SIAM International Conference on Data Mining, pp. 623-631 (2013).
19. Liu, Z., Hauskrecht, M.: Clinical time series prediction: Toward a hierarchical dynamical system framework. Artificial intelligence in medicine, vol. 65, pp. 5-18 (2015).

20. ipton, Z. C., Kale, D. C., Elkan, C., et al.: Learning to diagnose with LSTM recurrent neural networks. arXiv preprint arXiv:1511.03677 (2015).
21. Song, H., Rajan, D., Thiagarajan, J. J., et al.: Attend and diagnose: Clinical time series analysis using attention models. In: Thirty-second AAAI conference on artificial intelligence (2018).
22. Tsai, Y. H. H., Liang, P. P., et al.: Learning factorized multimodal representations. arXiv preprint arXiv:1806.06176 (2018).
23. Baltrušaitis, T., Ahuja, C., Morency, L. P.: Multimodal machine learning: A survey and taxonomy. IEEE transactions on pattern analysis and machine intelligence, vol. 41, pp. 423-443 (2018).
24. Xu, Y., Biswal, S., Deshpande, S. R., et al.: Raim: Recurrent attentive and intensive model of multimodal patient monitoring data. In: Proceedings of the 24th ACM SIGKDD international conference on Knowledge Discovery & Data Mining, pp. 2565-2573 (2018).
25. Rajkomar, A., Oren, E., Chen, K., et al.: Scalable and accurate deep learning with electronic health records. NPJ Digital Medicine, vol. 1, pp. 1-10 (2018).
26. Zhao, Y., Hong, Q., Zhang, X., et al.: BERTSurv: BERT-Based Survival Models for Predicting Outcomes of Trauma Patients. arXiv preprint arXiv:2103.10928 (2021).
27. Vaswani, A., Shazeer, N., Parmar, N., et al.: Attention is all you need. In: Advances in neural information processing systems, pp. 5998-6008 (2017).
28. Huang, K., Altosaar, J., & Ranganath, R.: Clinicalbert: Modeling clinical notes and predicting hospital readmission. arXiv preprint arXiv:1904.05342 (2019).
29. Kim, Y.: Convolutional Neural Networks for Sentence Classification. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1746-1751 (2014).
30. Trask, A., Gilmore, D., Russell, M.: Modeling order in neural word embeddings at scale. In: International Conference on Machine Learning, pp. 2266-2275 (2015).
31. Harutyunyan, H., Khachatrian, H., Kale, D. C., et al.: Multitask learning and benchmarking with clinical time series data. Scientific data, vol. 6, pp. 1-18 (2019).
32. Angus, D. C., Linde-Zwirble, W. T., Lidicker, J., et al.: Epidemiology of severe sepsis in the United States: analysis of incidence, outcome, and associated costs of care. Critical care medicine, vol. 29, pp. 1303-1310 (2001).
33. Paszke, A., Gross, S., Massa, F., et al.: Pytorch: An imperative style, high-performance deep learning library. In: Advances in neural information processing systems, pp. 8026-8037 (2019).
34. Kingma, D. P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014).
35. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation, vol. 9, pp. 1735-1780 (1997).
36. Schuster, M., Paliwal, K. K.: Bidirectional recurrent neural networks. IEEE transactions on Signal Processing, vol. 45, pp. 2673-2681 (1997).
37. Cho, K., Van Merriënboer, B., Bahdanau, D., et al.: On the properties of neural machine translation: Encoder-decoder approaches. arXiv preprint arXiv:1409.1259 (2014).
38. Bojanowski, P., Grave, E., Joulin, A., et al.: Enriching word vectors with subword information. Transactions of the Association for Computational Linguistics, vol. 5, pp. 135-146 (2017).
39. Peters, M. E., Neumann, M., Iyyer, M., et al.: Deep contextualized word representations. In: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, vol. 1, pp. 2227-2237 (2018).

Enhancing Transformer Efficiency for Multivariate Time Series Classification

Yuqing Wang[1], Yun Zhao[1], and Linda Petzold[1]

Department of Computer Science, University of California, Santa Barbara
wang603@ucsb.edu

Abstract. Most current multivariate time series (MTS) classification algorithms focus on improving the predictive accuracy. However, for large-scale (either high-dimensional or long-sequential) time series (TS) datasets, there is an additional consideration: to design an efficient network architecture to reduce computational costs such as training time and memory footprint. In this work we propose a methodology based on module-wise pruning and Pareto analysis to investigate the relationship between model efficiency and accuracy, as well as its complexity. Comprehensive experiments on benchmark MTS datasets illustrate the effectiveness of our method.

Keywords: Deep learning · Model efficiency · Pareto analysis · Time series classification.

1 Introduction

Time series (TS) data is ubiquitous, occurring in healthcare [1], stock market [2], astronomy [3], and many other domains [4,5]. With the advance of sensing techniques, TS classification across wide-ranging domains has gained much interest during the past decade [6,7].

The availability of the UCR/UEA time series benchmark datasets [7] has led to an abundance of TS classification algorithms [8,9,10,11,12]. The classification accuracy has been the key metric used to evaluate existing methods [13]. However, the high accuracy of these algorithms often comes with the cost of high computational complexity [14]. From common preconceptions in natural language processing (NLP) and computer vision (CV), in order to achieve high accuracy, training top performing models with millions/billions of parameters is a computationally intensive task, requiring days or weeks on many parallel GPUs or TPUs. However, such intensive training makes the model difficult to retrain for further improvement on performance. Likewise, for large-scale time series data with high dimensionality or long sequence length, it is challenging to maintain the balance between the predictive accuracy and training efficiency.

In this work, we propose a method to investigate the relationship between model efficiency and its effectiveness, as well as its complexity for MTS classification. The model architecture is based on Transformer and Fourier transform. We use 18 benchmark MTS datasets for evaluation. Comprehensive experiments

are conducted on all datasets, including ablation study of each module of the network and module-by-module pruning in terms of accuracy, training speed, and model size. Experimental results demonstrate the competitive performance of our proposed architecture compared with current state-of-the-art methods. Ablation studies identify the main contributors to the predictive performance, such as multi-head self-attention and Fourier transform. In addition, module-wise pruning of the network reveals the trade-off between model efficiency and effectiveness, as well as model efficiency and complexity. Finally, we conduct Pareto analysis to examine the trade-off between efficiency and performance.

The main contributions of this paper are highlighted as follows:

(1) To the best of our knowledge, this is the first paper to perform Pareto analysis to investigate the relationship between efficiency and accuracy.
(2) Through module-by-module pruning, comprehensive experimental results indicate an evident trade-off between model efficiency and its effectiveness, as well as its complexity.
(3) We employ Pareto analysis to investigate the relationship between model efficiency and performance. Such analysis methods can provide general guidance for researchers on how to select efficient model configurations, which can be applied to any model architecture.

The remainder of this paper is organized as follows. Section 2 describes related work of Transformer and Fourier transform on time series analysis and existing methods on model efficiency improvement. The network architecture is outlined in Section 3. Section 4 discusses datasets and experiments on 18 benchmark datasets, including ablation studies, module-wise pruning and Pareto efficiency visualization. Finally, our conclusions are presented in Section 5.

2 RELATED WORK

Neural Networks for Time Series Classification. Currently, most TS classification algorithms can be divided into three categories: feature-based [15], distance-based [16], and neural network based methods [6]. Here, we focus only on neural network based methods. Since the advancements of deep learning, two popular frameworks, CNN and RNN, are widely applied in TS classification tasks. [17] combined Fully Convolutional Networks (FCN) and Residual Networks (ResNet) for univariate time series classification. [18] developed a group-constrained method, which combines a CNN with an RNN. More recent works such as InceptionTime [19], TapNet [20], and TST [12] are proposed for TS classification. For additional deep learning methods, we refer readers to [6].

Fourier Transform in Time Series. The Fourier transform (FT) has been an important tool in time series analysis for decades [21], and is widely used for applications such as anomaly detection [22], periodicity detection [23], and similarity measures [24]. The FT converts a TS from time domain to frequency domain, and uses Fourier coefficients to represent the original data. For the TS

classification task, FTs have been used indirectly in disparate applications. For instance, [25] utilizes the FT to filter noisy data for vegetation type classification, and [26] uses the FT as a feature extraction technique to classify electroencephalography (EEG) data. However, none of the above methods apply the FT directly to TS classification, particularly in the context of neural networks. In contrast, we aim to apply the discrete FT and its inverse as modules of a deep learning framework. The unparameterized FT can reduce the computational cost of the network to some extent.

Transformer Networks for Time Series Classification. With the exemplary performance of the Transformer architecture [27] in NLP and CV, researchers in the time series community began exploring Transformers in TS classification in specific domains [28,29]. More recent works have generalized Transformer frameworks for MTS classification. [12] adopts a Transformer encoder architecture for unsupervised representation learning of MTS. [30] explored an extension of the current Transformer architecture by gating, which merges two towers for MTS classification. In contrast, we propose to generalize a mixing framework which utilizes both Transformer and FT. By replacing some self-attention sublayers with FT, the computational complexity can be reduced.

Model Training Efficiency. Due to the increasing size of both models and training data, many works have focused on improving model training efficiency through parameter reduction, such as DenseNet [31] and EfficientNet [32], training speed improvement including NFNets [33] and BotNet [34], or both [35]. One of the most common techniques to improve network efficiency is model pruning. Early works focused on non-structured methods. For instance, [36,37] proposed to remove individual weight values. Recent works focused more on structured methods, such as channel weight pruning based on l_1 norm [38].

3 METHODOLOGY

In this section, we present our network architecture, which contains all of the modules for potential model pruning. The overall model structure is illustrated in Figure 1.

Input Embeddings. Input embeddings are commonly used in NLP models, which map relatively low-dimensional vectors to high-dimensional vectors to facilitate sequence modeling [39]. Correspondingly, an embedding for TS sequence is required to capture the dependencies among different features without considering the temporal information [40]. Our framework employs a 1D convolutional layer to obtain the K-dimensional embeddings at each time step.

Discrete Fourier Transform. The Fourier transform decomposes a function of time into its constituent frequencies. For clarity, we first consider the 1D Discrete Fourier transform (DFT). Given a sequence of complex numbers $x(n)$

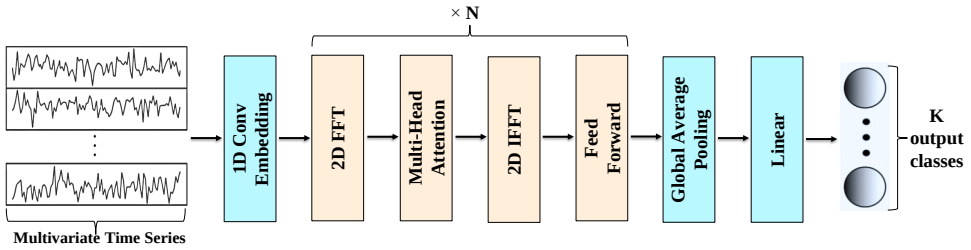


Fig. 1: An overview of the full model framework. Our architecture is based on Transformer and Fourier transform. Following the sequence embedding, we apply a 2D discrete Fourier transform (particularly Fast Fourier transform) to convert the TS features from the time domain to the frequency domain, a multi-head self-attention layer, and a 2D inverse discrete Fourier transform to map the features back to the time domain. Then we employ a Global Average Pooling (GAP) layer to average the output of the MTS over the entire time dimension. Finally, a Softmax layer is used for the multi-class MTS classification task.

with $0 \leq n \leq N-1$, the 1D DFT is defined by

$$X(k) = \sum_{n=0}^{N-1} x(n) \cdot e^{-\frac{2\pi i}{N} kn} = \sum_{n=0}^{N-1} x(n) \cdot W_N^{kn}, \ 0 \leq k \leq N-1,$$

where $W_N^{kn} = e^{-\frac{2\pi i}{N} kn}$. Given the DFT $X(k)$, the original sequence can be recovered by the inverse DFT (IDFT)

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) \cdot e^{\frac{2\pi i}{N} kn}, \ 0 \leq n \leq N-1.$$

The 2D DFT is a direct extension of the 1D DFT, obtained by alternately performing the 1D DFT on the row and column dimensions. Given a 2D signal $x(m,n)$ with $0 \leq m \leq M-1, 0 \leq n \leq N-1$, the 2D DFT is given by

$$X(k,l) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} x(m,n) \cdot e^{-2\pi j(\frac{km}{M} + \frac{ln}{N})}.$$

Similar to the 1D IDFT, the 2D DFT is invertible via the 2D IDFT,

$$x(m,n) = \frac{1}{MN} \sum_{k=0}^{M-1} \sum_{l=0}^{N-1} X(k,l) \cdot e^{2\pi j(\frac{km}{M} + \frac{ln}{N})}.$$

To compute the DFT efficiently, the Fast Fourier Transform (FFT) algorithm takes advantage of the periodicity and symmetry properties of W_N^{kn} such that

the computational complexity of the DFT reduces from $O(N^2)$ to $O(NlogN)$, regardless of dimension.

Multi-head Attention. The multi-head attention (MHA) mechanism, the major component of the Transformer architecture [27], allows the model to jointly attend to information from different representation subspaces at different positions. MHA is defined as:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \text{head}_2, \cdots, \text{head}_h)W^O,$$

where $Q, K, V \in \mathbb{R}^{n \times d_{model}}$ are input embedding matrices, n is the sequence length, d_{model} is the embedding dimension, and h is the number of heads. Each head i is defined as:

$$\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V) = \text{softmax}\left(\frac{QW_i^Q(KW_i^K)^T}{\sqrt{d_k}}\right)VW_i^V,$$

where $W_i^Q \in \mathbb{R}^{d_{model} \times d_k}, W_i^K \in \mathbb{R}^{d_{model} \times d_k}, W_i^V \in \mathbb{R}^{d_{model} \times d_v}, W_i^O \in \mathbb{R}^{hdv \times d_{model}}$ are parameter matrices to be learned.

Global Average Pooling. Global average pooling involves calculating the average value of all of the elements in a feature map. It is mainly used to reduce the amount of learnable parameters.

Batch Normalization. Instead of using layer normalization in Transformer-related architectures in NLP, we consider the necessity of applying batch normalization to each block shown in Figure 1. Compared to layer normalization, batch normalization can mitigate the effect of outlier values in time series data, which does not appear in text representations.

Activation Function. Using the same activation function as the original Transformer architecture [27], we consider the necessity of applying the activation function *gelu* for each module shown in Figure 1.

Feedforward Neural Network. A position-wise feedforward neural network (FNN) is applied with two 1D convolutional layers with kernel size 1, and a *gelu* activation function in between.

4 EXPERIMENTS

In this section, we describe benchmark MTS datasets [7] used for experimental evaluation, the experimental setup, and corresponding results.

4.1 DATASETS

We select a set of 18 publicly available benchmark datasets from the UCR/UEA classification archive: AtrialFibrillation (AF), BasicMotions (BM), Cricket (CR), DuckDuckGeese (DDG), Epilepsy (EP), EthanolConcentration (EC), ERing (ER), FingerMovements (FM), HandMovementDirection (HMD), Handwriting (HW), Heartbeat (HB), Libras (LIB), NATOPS (NATO), PEMS-SF (PEMS), RacketSports (RS), SelfRegulationSCP1 (SRS1), SelfRegulationSCP2 (SRS2),

and UWaveGestureLibrary (UW). The main characteristics of each dataset are summarised in Table 1. All of the datasets have been split into training and testing sets by default. Thus, there are no preprocessing steps for these data. The predictive performance on all datasets is evaluated in terms of accuracy.

Table 1: Summary of the 18 UCR/UEA datasets used in experimentation.

Dataset	Code	Train Size	Test Size	Dimensions	Length	Classes
AtrialFibrillation	AF	15	15	2	640	3
BasicMotions	BM	40	40	6	100	4
Cricket	CR	108	72	6	1197	12
DuckDuckGeese	DDG	50	50	1345	270	5
Epilepsy	EP	137	138	3	206	4
EthanolConcentration	EC	261	263	3	1751	4
ERing	ER	30	270	4	65	6
FingerMovements	FM	316	100	28	50	2
HandMovementDirection	HMD	160	74	10	400	4
Handwriting	HW	150	850	3	152	26
Heartbeat	HB	204	205	61	405	2
Libras	LIB	180	180	2	45	15
NATOPS	NATO	180	180	24	51	6
PEMS-SF	PEMS	267	173	963	144	7
RacketSports	RS	151	152	6	30	4
SelfRegulationSCP1	SRS1	268	293	6	896	2
SelfRegulationSCP2	SRS2	200	180	7	1152	2
UWaveGestureLibrary	UW	120	320	3	315	8

4.2 SETUP

We set aside 20% of the default training set for the validation set, which we used to select the best collection of hyperparameters. All experiments were implemented in Pytorch [41] on one GTX 1080 Ti GPU. We minimized the cross entropy loss with the Adam [42] optimizer for training. The hyperparameter search space for each dataset is listed in Table 2. Note that the batch size choice is limited by the available GPU memory.

Table 2: Hyperparameter search space of the model on each dataset. If the number of layers of a module is equal to 0, then this module is removed in the pruned model.

Hyperparameters	Search Space
learning rate	[1e-3, 5e-3, 1e-4, 5e-4, 1e-5, 5e-5]
dropout rate	[0.1, 0.2, 0.3]
batch size	[8, 16, 32]
♯ of heads	[4, 8, 16]
♯ of FFT layers	[0, 1, 2, 3, 4]
♯ of IFFT layers	[0, 1, 2, 3, 4]
♯ of MHA layers	[0, 1, 2, 3, 4]
♯ of Feedforward layers	[0, 1, 2, 3, 4]

Table 3: Ablation study in the testing accuracy loss on 18 datasets by removing each module at a time while leaving others the same. Each experiment is conducted 5 times with different random seeds. The results are shown in the format of mean and standard deviation. Column 2 shows the accuracy of the full model with all modules included. Columns 3 to 10 represent the accuracy when the module in that column is removed from the model. Bold indicates that the module contributes most to the loss in accuracy and underlining indicates that the module contributes least to the loss in accuracy when the module is removed.

Dataset	Acc.	Unpruned	EMBED	FFT	IFFT	MHA	FFN	GAP	BN	ACT
AF	Mean	0.667	0.600	**0.400**	0.467	**0.400**	0.667	0.533	0.600	0.667
	Std.	0.003	0.005	0.005	0.004	0.003	0.006	0.006	0.004	0.003
BM	Mean	0.975	0.950	**0.725**	0.775	0.750	0.900	0.925	0.900	0.950
	Std.	0.008	0.010	0.012	0.009	0.012	0.010	0.014	0.009	0.011
CR	Mean	0.987	0.958	0.875	0.861	**0.833**	0.889	0.944	0.972	0.944
	Std.	0.007	0.009	0.012	0.008	0.012	0.006	0.009	0.012	0.008
DDG	Mean	0.580	0.580	0.440	0.420	**0.380**	0.520	0.560	0.560	0.580
	Std.	0.016	0.017	0.020	0.016	0.014	0.016	0.016	0.014	0.016
EP	Mean	0.986	0.978	**0.891**	0.913	0.899	0.949	0.971	0.956	0.971
	Std.	0.014	0.013	0.016	0.014	0.014	0.012	0.014	0.013	0.015
EC	Mean	0.456	0.445	0.376	0.395	**0.365**	0.418	0.441	0.445	0.452
	Std.	0.003	0.002	0.003	0.003	0.004	0.002	0.004	0.003	0.002
ER	Mean	0.963	0.956	0.896	0.889	**0.885**	0.892	0.948	0.952	0.956
	Std.	0.006	0.007	0.006	0.006	0.008	0.005	0.006	0.007	0.005
FM	Mean	0.640	0.620	**0.490**	0.520	0.500	0.600	0.590	0.610	0.620
	Std.	0.009	0.008	0.007	0.008	0.010	0.008	0.009	0.010	0.011
HMD	Mean	0.486	0.446	0.365	0.351	**0.338**	0.406	0.459	0.432	0.473
	Std.	0.018	0.016	0.020	0.017	0.018	0.019	0.018	0.016	0.020
HW	Mean	0.529	0.514	0.471	0.473	**0.468**	0.506	0.506	0.512	0.514
	Std.	0.006	0.007	0.006	0.005	0.007	0.007	0.008	0.007	0.006
HB	Mean	0.771	0.766	**0.683**	0.707	0.688	0.751	0.756	0.766	0.756
	Std.	0.014	0.015	0.014	0.017	0.015	0.016	0.014	0.015	0.016
LIB	Mean	0.917	0.906	**0.822**	0.827	0.839	0.889	0.894	0.906	0.911
	Std.	0.009	0.011	0.012	0.010	0.012	0.013	0.011	0.009	0.010
NATO	Mean	0.844	0.833	**0.728**	0.739	0.750	0.772	0.811	0.833	0.833
	Std.	0.005	0.004	0.005	0.007	0.006	0.005	0.006	0.004	0.006
PEMS	Mean	0.908	0.884	0.815	0.809	**0.803**	0.867	0.879	0.896	0.896
	Std.	0.013	0.012	0.014	0.016	0.014	0.013	0.013	0.014	0.012
RS	Mean	0.914	0.901	**0.796**	0.816	0.803	0.855	0.908	0.901	0.908
	Std.	0.021	0.020	0.020	0.018	0.019	0.021	0.020	0.021	0.019
SRS1	Mean	0.915	0.894	0.836	0.823	**0.819**	0.853	0.887	0.894	0.901
	Std.	0.005	0.007	0.006	0.006	0.005	0.007	0.006	0.005	0.005
SRS2	Mean	0.600	0.594	**0.522**	0.533	0.516	0.578	0.583	0.588	0.594
	Std.	0.002	0.003	0.002	0.001	0.004	0.002	0.003	0.003	0.002
UW	Mean	0.922	0.906	0.844	0.850	**0.841**	0.875	0.894	0.897	0.903
	Std.	0.006	0.008	0.009	0.006	0.007	0.008	0.006	0.007	0.007

4.3 MODULE SETTINGS

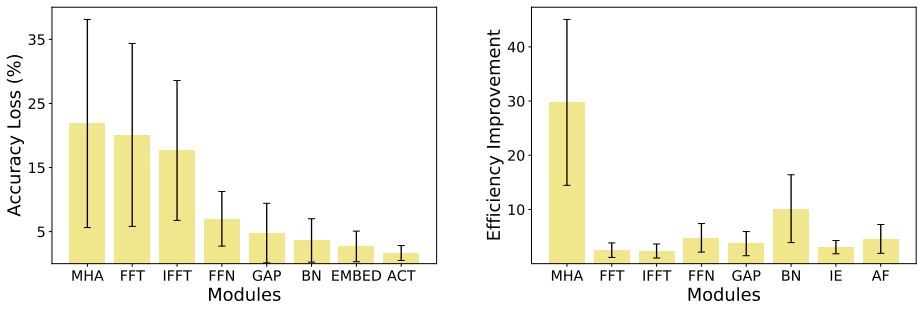
Based on Section 3, we define the following eight modules of the network for further analysis: input embedding (EMBED), fast Fourier transform (FFT), inverse fast Fourier transform (IFFT), multi-head attention (MHA), feedforward neural network (FFN), global average pooling (GAP), batch normalization (BN), and activation function (ACT). The corresponding abbreviations of each module are shown in parentheses.

4.4 ABLATION STUDY

First, we conduct ablation studies to analyze the contributions of each module on the predictive performance. The contribution of each module is obtained when a module is removed from the full network while other modules remain intact. The fine-tuned results on 18 datasets are shown in Table 3. Starting from Column 4, the smaller the accuracy is, the larger the module's contribution is, and vice versa. The accuracy of each dataset for the unpruned model (Table 3 Column 3) is competitive with current state-of-the-art methods [7]. Among eight modules, it can be seen that MHA and FFT contribute most to the predictive performance on 10 out of the 18 datasets and 9 out of the 18 datasets, respectively. For MTS data, the correlations between different dimensions across all time steps are important to consider. Hence, the MHA is able to catch different feature correlations, and influence the accuracy to a large extent. The FFT, as the core of signal processing and more generalized time series, extracts frequency information embedded in data, which provides a more straightforward representation compared to the original data in the time-domain. In contrast, we observe that EMBED, BN, and ACT contribute least to the predictive performance on 11 out of the 18 datasets, 5 out of the 18 datasets, and 13 out of the 18 datasets, respectively. Although these operations are important for the training of the model, they influence the testing accuracy marginally compared with MHA and FFT.

To clearly demonstrate the influence of each module on the predictive performance and efficiency of the network, the averaged testing accuracy loss and the corresponding efficiency improvement for each module (compared with the unpruned model) over all datasets are presented in Figure 2. Here, efficiency is defined as the product of training time per epoch and the amount of learnable parameters. The higher the product, the lower the efficiency is. In consideration of highly diversed datasets with respect to sequence length, number of samples, and dimensionality, the average loss in accuracy for each module demonstrates a high variance from Figure 2a as the performance loss extent can vary depending on dataset characteristics. The modules MHA, FFT, and IFFT demonstrate a notable influence on the model performance on average (21.9%, 20.1%, and 17.7% loss in accuracy respectively). For modules like BN, EMBED and ACT, removing them bring about minimal accuracy loss compared to other modules (3.6%, 2.7%, and 1.6% respectively). Meanwhile, comparing Figure 2a and Figure 2b, the module which has larger impact on the predictive performance does not indicate that removing it can bring about more efficiency improvement. For instance,

the computationally inexpensive FFT influences the predictive performance to a large extent. In contrast, although the computational cost of BN is high, its contribution to the performance is marginal.



(a) (b)

Fig. 2: (a) represents the average testing accuracy loss across all datasets while removing one module at a time and other modules remain in the network. Modules MHA, FFT, and IFFT bring about larger influence on the predictive performance due to the high percentage of accuracy loss when removing them. In comparison, BN, EMBED, and ACT bring about marginal influence on the predictive performance compared with other modules. (b) represents the corresponding average efficiency improvement across all datasets when one module is removed from the network while other modules keep intact.

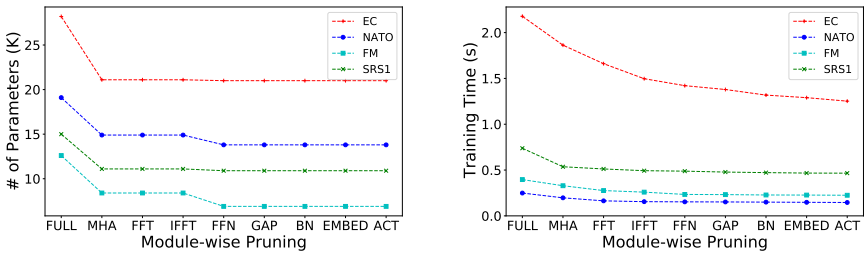
4.5 MODULE-BY-MODULE PRUNING

Next, we explore the relationship between efficiency (defined the same as Section 4.4) and effectiveness (predictive performance). Based on the contribution of each module on the performance loss shown in Figure 2a, we perform module-by-module pruning by following the order of modules from the most significant contributor to the least significant contributor (MHA, FFT, IFFT, FFN, GAP, BN, EMBED, ACT) to accuracy. We evaluate such pruning effect in two aspects: (1) effectiveness: testing accuracy; (2) efficiency: average training time per epoch in seconds and the number of learnable parameters. Due to limited space, we only show some datasets' testing accuracy in Table 4 and their efficiency results in Figure 3. We observe that after removing the entire MHA module, the number of learnable parameters shrinks drastically, so as the accuracy (Table 4 Column 4). The representation capability of the pruned network, which has fewer parameters, is damaged since the amount of parameters is a key aspect to the network representation. Furthermore, the pace of accuracy loss and parameter reduction removal of subsequent modules slows down as FFT/IFFT has no learnable parameters. For the remaining modules, the number of parameters they carry is

much fewer than the MHA module. Based on Figure 2a, their effects on the predictive performance are moderate. Hence, the curves in Figure 3 are relatively flat following MHA. We further investigate the extent of change in accuracy of module-wise pruning on all datasets, as shown in Figure 4. We notice that the performance variation in different datasets vary widely. For datasets such as AF, BM, and DDG, the model pruning has a great impact on their performance. This may be due to very limit amount of training samples. Conversely, for datasets like HB, LIB, and SRS1, the model pruning brings little effect after removing the MHA module (within 1%).

Table 4: Module-wise pruning results of datasets EC, NATO, FM and SRS1. The results from Column 3 (MHA) to Column 10 (AF) with regard to accuracy represent that the module in that column is removed from the model architecture. Experiments are conducted 5 times with different random seeds. The accuracy results are shown in the format of mean and standard deviation. Bold represents that the module brings about much accuracy loss compared to the unpruned model. Following MHA, the accuracy decreasing trend remains stable.

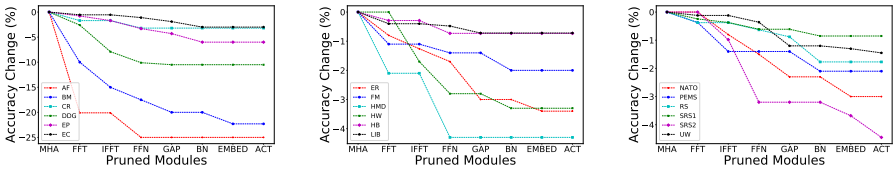
Dataset	Acc.	Unpruned	MHA	FFT	IFFT	FFN	GAP	BN	IE	AF
EC	Mean	0.456	**0.365**	0.363	0.363	0.361	0.358	0.354	0.354	0.354
	Std.	0.003	0.004	0.004	0.002	0.004	0.003	0.003	0.003	0.003
NATO	Mean	0.844	**0.750**	0.750	0.744	0.739	0.733	0.733	0.728	0.728
	Std.	0.005	0.006	0.003	0.004	0.006	0.005	0.006	0.004	0.005
FM	Mean	0.640	**0.500**	0.495	0.495	0.493	0.493	0.490	0.490	0.490
	Std.	0.009	0.010	0.011	0.010	0.008	0.009	0.011	0.010	0.011
SRS1	Mean	0.915	**0.819**	0.817	0.816	0.814	0.814	0.812	0.812	0.812
	Std.	0.005	0.005	0.003	0.003	0.0.004	0.006	0.003	0.004	0.005



(a) (b)

Fig. 3: Module-wise results for changes in terms of number of parameters and training time per epoch on four datasets: EC, NATO, FM, SRS1.

Overall, based on the above module-by-module pruning scheme, we observe that as the effectiveness (predictive performance) of the network increases, the



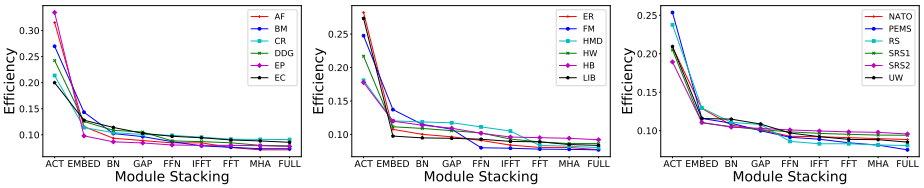
(a) (b) (c)

Fig. 4: Change in accuracy (%) from module-by-module pruning across all datasets. The order of datasets shown from (a) to (c) correspond to Table 3.

corresponding efficiency (training speed and model size) generally decreases. The evident cost–benefit trade-off between efficiency and effectiveness provides a key question to researchers on how to find efficient model settings while maintaining the "equilibrium" between these two aspects. This problem will be discussed in Section 4.7.

4.6 EFFICIENCY VS. COMPLEXITY

Here, we explore the relationship between network efficiency and complexity. In general, the more complex a model is, the less efficient it is. The network's efficiency is defined in the same way as previous sections, in terms of the training time and the number of parameters. Meanwhile, we define the complexity of the model as the stacking of modules. Contrary to model pruning, we stack each module based on their influence on the predictive performance, from the least significant contributor to the most significant contributor (ACT, EMBED, BN, GAP, FFN, IFFT, FFT, MHA) to accuracy. Our empirical results in Figure 5 shed light on the trade-off between model efficiency and complexity. As can be seen in Figure 5, as more modules are stacked over the network, the corresponding computational efficiency decreases. All datasets illustrate similar trends.

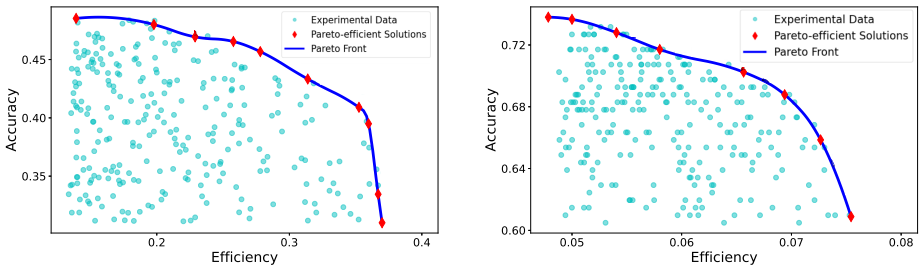


(a) (b) (c)

Fig. 5: Trade-off between network efficiency and complexity across all datasets. Due to the notable differences of dataset sizes, the computation of efficiency is normalized for each dataset. The order of datasets shown from (a) to (c) correspond to Table 3.

4.7 PARETO ANALYSIS FOR TRADE-OFF EXPLORATION BETWEEN EFFICIENCY AND PERFORMANCE/EFFECTIVENESS

We define the model efficiency in terms of the reciprocal of the product between training time per epoch and the number of parameters. Thus, the higher the reciprocal, the higher the efficiency. To explore the relationship between model efficiency and performance, we employ Pareto analysis [43]. Pareto efficiency represents a state for which improving the performance as measured by one criterion would worsen the performance as measured by another criterion. We choose the *FingerMovements* and *Heartbeat* datasets to obtain the Pareto frontiers, where the set of points on the front correspond to Pareto-efficient solutions. We have two objectives: (1) maximize the efficiency; (2) maximize the accuracy. Figure 6 shows the result of Pareto fronts for both datasets in blue, where the red points are Pareto-efficient solutions. The scattered cyan points are randomly sampled experimental data from all different configurations. The Pareto analysis provides us with a principled approach for choosing efficient network settings, while exploring the trade-off between efficiency and performance. Specifically, we can identify the extent of computational resources that is required in order for a model to achieve a certain performance. Conversely, we can identify how well a model can perform, given a certain amount of resources.



(a) *FingerMovements*. (b) *Heartbeat*.

Fig. 6: Pareto efficiency visualization of the *FingerMovements* and *Heartbeat* datasets. The scattered cyan points, the marked red points, and the blue curve represent randomly sampled experimental data, Pareto-efficient solutions, and Pareto efficient frontiers.

5 DISCUSSION

In this work, we propose a methodology to investigate the relationship between model efficiency and effectiveness, as well as its complexity. The method is performed on a mixing network based on Transformer and Fourier transform for

MTS classification. Extensive experiments are conducted on 18 MTS datasets, including ablation studies on different modules of the network, module-by-module pruning evaluated in terms of the predictive performance, training speed, and the number of learnable parameters. The network achieves competitive performance compared to current best-performing methods. Ablation studies indicate that self-attention and Fourier transform are the largest contributors that influence the model performance across all datasets. Furthermore, through sequential pruning of each module, we observed the efficiency–effectiveness and the efficiency–complexity trade-offs of the network. Through Pareto analysis, we show how to choose efficient settings of the network, while investigating the performance–efficiency trade-off through visualization of the Pareto fronts. We note that for far more complex models applied to large-scale data, due to finite computational resources, it is not practical to consider all possible configurations of the model and perform experiments. In these cases, given a reasonable number of experiments, techniques like regression can be used to generate massive random model settings and corresponding model performance. Pareto analysis can then be performed to evaluate the efficiency-performance trade-off, to guide researchers to adjust the model settings to improve the efficiency and effectiveness accordingly.

References

1. Li-wei, H. L., Adams, R. P., Mayaud, L., et al.: A physiological time series dynamics-based approach to patient monitoring and outcome prediction. IEEE journal of biomedical and health informatics, vol. 19, pp. 1068-1076 (2014).
2. Liu, H., Long, Z.: An improved deep learning model for predicting stock market price time series. Digital Signal Processing, vol. 102, pp. 102741 (2020).
3. Fu, T. C.: A review on time series data mining. Engineering Applications of Artificial Intelligence, vol. 24, pp. 164-181 (2011).
4. Gao, B., Li, X., Woo, W. L., et al.: Physics-based image segmentation using first order statistical properties and genetic algorithm for inductive thermography imaging. IEEE Transactions on Image Processing, vol. 27, pp. 2160-2175 (2017).
5. Hu, B., Gao, B., Woo, W. L., et al.: A Lightweight Spatial and Temporal Multi-Feature Fusion Network for Defect Detection. IEEE Transactions on Image Processing, vol. 30, pp. 472-486 (2020).
6. Fawaz, H. I., Forestier, G., Weber, J., et al.: Deep learning for time series classification: a review. Data mining and knowledge discovery, vol. 33, pp. 917-963 (2019).
7. Ruiz, A. P., Flynn, M., Large, J., et al.: The great multivariate time series classification bake off: a review and experimental evaluation of recent algorithmic advances. Data Mining and Knowledge Discovery, vol. 35, pp. 401-449 (2021).
8. Hüsken, M., Stagge, P.: Recurrent neural networks for time series classification. Neurocomputing, vol. 50, pp. 223-235 (2003).
9. Zhao, B., Lu, H., Chen, S., et al.: Convolutional neural networks for time series classification. Journal of Systems Engineering and Electronics, vol. 28, pp. 162-169 (2017).
10. Lines, J., Taylor, S., Bagnall, A.: Time series classification with HIVE-COTE: The hierarchical vote collective of transformation-based ensembles. ACM Transactions on Knowledge Discovery from Data, vol. 12 (2018).

11. Dempster, A., Petitjean, F., Webb, G. I.: ROCKET: exceptionally fast and accurate time series classification using random convolutional kernels. Data Mining and Knowledge Discovery, vol. 34, pp. 1454-1495 (2020).
12. Zerveas, G., Jayaraman, S., Patel, D., et al.: A transformer-based framework for multivariate time series representation learning. In: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, pp. 2114-2124 (2021).
13. Lines, J., Bagnall, A.: Time series classification with ensembles of elastic distance measures. Data Mining and Knowledge Discovery, vol. 25, pp. 565-592 (2015).
14. Schäfer, P.: Scalable time series classification. Data Mining and Knowledge Discovery, vol. 30, pp. 1273-1298 (2016).
15. Fulcher, B. D., Jones, N. S.: Highly comparative feature-based time-series classification. IEEE Transactions on Knowledge and Data Engineering, vol. 26, pp. 3026-3037 (2014).
16. Abanda, A., Mori, U., Lozano, J. A.: A review on distance based time series classification. Data Mining and Knowledge Discovery, vol. 33, pp. 378-412 (2019).
17. Wang, Z., Yan, W., Oates, T.: Time series classification from scratch with deep neural networks: A strong baseline. In: 2017 International joint conference on neural networks, pp. 1578-1585 (2017).
18. Lin, S., Runger, G. C.: GCRNN: Group-constrained convolutional recurrent neural network. IEEE transactions on neural networks and learning systems, vol. 29, pp.4709-4718 (2017).
19. Fawaz, H. I., Lucas, B., Forestier, G., et al.: Inceptiontime: Finding alexnet for time series classification. Data Mining and Knowledge Discovery, vol. 34, pp. 1936-1962 (2020).
20. Zhang, X., Gao, Y., Lin, J., et al.: Tapnet: Multivariate time series classification with attentional prototypical network. In: Proceedings of the AAAI Conference on Artificial Intelligence, pp. 6845-6852 (2020).
21. Bloomfield, P.: Fourier analysis of time series: an introduction (2004).
22. Ren, H., Xu, B., Wang, Y., et al.: Time-series anomaly detection service at microsoft. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 3009-3017 (2019).
23. Puech, T., Boussard, M., D'Amato, A., et al.: A fully automated periodicity detection in time series. In: International Workshop on Advanced Analysis and Learning on Temporal Data, pp. 43-54 (2019).
24. Janacek, G. J., Bagnall, A. J., Powell, M.: A likelihood ratio distance measure for the similarity between the fourier transform of time series. In: Pacific-Asia Conference on Knowledge Discovery and Data Mining, pp. 737-743 (2005).
25. Geerken, R., Zaitchik, B., Evans, J. P.: Classifying rangeland vegetation type and coverage from NDVI time series using Fourier Filtered Cycle Similarity. International Journal of Remote Sensing, vol. 26, pp. 5535-5554 (2005).
26. Samiee, K., Kovacs, P., Gabbouj, M.: Epileptic seizure classification of EEG time-series using rational discrete short-time Fourier transform. IEEE transactions on Biomedical Engineering, vol. 62, pp. 541-552 (2014).
27. Vaswani, A., Shazeer, N., Parmar, N., et al.: Attention is all you need. In: Advances in neural information processing systems, pp. 5998-6008 (2017).
28. Oh, J., Wang, J., Wiens, J.: Learning to exploit invariances in clinical time-series data using sequence transformer networks. In: Machine Learning for Healthcare Conference, pp. 332-347 (2018).
29. Zhao, Y., Hong, Q., Zhang, X., et al.: BERTSurv: BERT-Based Survival Models for Predicting Outcomes of Trauma Patients. arXiv preprint arXiv:2103.10928 (2021).

30. Liu, M., Ren, S., Ma, S., et al.: Gated Transformer Networks for Multivariate Time Series Classification. arXiv preprint arXiv:2103.14438 (2021).
31. Huang, G., Liu, Z., Van Der Maaten, L., et al.: Densely connected convolutional networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 4700-4708 (2017).
32. Tan, M., Le, Q.: Efficientnet: Rethinking model scaling for convolutional neural networks. In: International Conference on Machine Learning, pp. 6105-6114 (2019).
33. Brock, A., De, S., Smith, S. L., et al.: High-performance large-scale image recognition without normalization. arXiv preprint arXiv:2102.06171 (2021).
34. Srinivas, A., Lin, T. Y., Parmar, N., et al.: Bottleneck transformers for visual recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 16519-16529 (2021).
35. Tan, M., Le, Q. V.: Efficientnetv2: Smaller models and faster training. arXiv preprint arXiv:2104.00298 (2021).
36. LeCun, Y., Denker, J. S., Solla, S. A.: Optimal brain damage. In: Advances in neural information processing systems, pp. 598-605 (1990).
37. Han, S., Pool, J., Tran, J., et al.: Learning both weights and connections for efficient neural networks. arXiv preprint arXiv:1506.02626 (2015).
38. Li, H., Kadav, A., Durdanovic, I., et al.: Pruning filters for efficient convnets. arXiv preprint arXiv:1608.08710 (2016).
39. Kim, Y.: Convolutional Neural Networks for Sentence Classification. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1746-1751 (2014).
40. Song, H., Rajan, D., Thiagarajan, J. J., et al.: Attend and diagnose: Clinical time series analysis using attention models. In: Thirty-second AAAI conference on artificial intelligence (2018).
41. Paszke, A., Gross, S., Massa, F., et al.: Pytorch: An imperative style, high-performance deep learning library. In: Advances in neural information processing systems, pp. 8026-8037 (2019).
42. Kingma, D. P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014).
43. Censor, Y.: Pareto optimality in multiobjective problems. Applied Mathematics and Optimization, vol. 4, pp. 41-59 (1977).

Client Error Clustering Approaches in Content Delivery Networks (CDN)

Ermiyas Birihanu[1], Jiyan Mahmud[1], Péter Kiss,[1], Adolf Kamuzora[1], Wadie Skaf[1], Tomáš Horváth[1], Tamás Jursonovics,[2], Peter Pogrzeba,[2], and Imre Lendák[1]

[1] Telekom Innovation Laboratories, Data Science and Engineering Department, Faculty of Informatics, Eötvös Loránd University Pázmány Péter str. 1/A, 1117 Budapest, Hungary
{ermiyasbirihanu, jiyan, axx6v4, adolfnfsp, skaf, tomas.horvath lendak}@inf.elte.hu ,
home page: http://t-labs.elte.hu/
[2] Deutsche Telekom, Berlin, Germany

Abstract. Content delivery networks (CDNs) are the backbone of the Internet and are key in delivering high quality video on demand (VoD), web content and file services to billions of users. CDNs usually consist of hierarchically organized content servers positioned as close to the customers as possible. CDN operators face a significant challenge when analyzing billions of web server and proxy logs generated by their systems. The main objective of this study was to analyze the applicability of various clustering methods in CDN error log analysis. We worked with real-life CDN proxy logs, identified key features included in the logs (e.g., content type, HTTP status code, time-of-day, host) and clustered the log lines corresponding to different host types offering live TV, video on demand, file caching and web content. Our experiments were run on a dataset consisting of proxy logs collected over a 7-day period from a single, physical CDN server running multiple types of services (VoD, live TV, file). The dataset consisted of 2.2 billion log lines. Our analysis showed that CDN error clustering is a viable approach towards identifying recurring errors and improving overall quality of service.

Keywords: Content Delivery Network, Error Clustering, HTTP proxy logs, HTTP status codes.

1 Introduction and problem definition

Customers consuming live TV, video on demand (VoD) or file services are accustomed to a high quality of service which is made possible by content delivery networks (CDN). Modern CDNs usually offer their services via protocols built on top of the HyperText Transfer Protocol (HTTP).The key infrastructure components of CDNs are origin and edge/surrogate servers as well as the communication infrastructure connecting them to their customers. Origin servers contain

large volumes of movies, series and other multimedia content and are usually limited in number. Edge servers are more numerous and they are positioned closer to the customers and cache only the most relevant content due to their limited storage capacity.Various HTTP proxy solutions are used on edge/surrogate servers to cache and serve content to thousands of concurrent users. Software and infrastructure errors occur both on the customer and CDN side. Customer software in set-top boxes, smart televisions and mobile devices might contain bugs or issue erroneous commands.The CDN infrastructure might contain bottlenecks, be under cyber attack or just have partial hardware or software failures. Considering the large number of customers and ever increasing numbers of nodes in modern CDNs, the number of such errors is continuously increasing.

Log files can provide important insights about the condition of a system or device. These files provide information about whether or not a system is working properly, as well as how actions or services perform. Various types of information are stored in log files on different web servers such as username, timestamp, last page accessed, success rate, user agent, Universal Resource Locator (URL), etc. By analysing these log files, one can better understand user and system behaviour. Since complex system generate large quantities of log information, manual analysis is not practical; thus, automated approaches are warranted [1].

The goal of this research is to contribute to the state of the art by analyzing edge server logs collected at a European CDN provider. The error logs analyzed were clustered on the host (i.e., CDN node) level. The ultimate goal of this research was to identify the causes of the diverse errors, provide valuable insights to the operators and developers of the analyzed CDN and thereby contribute towards solving recurring problems and propose potential improvements.

2 Related works

The growing number of Internet users and their increasing demand for the delivery of low-latency content led to the emergence of networks of content delivery networks (CDN) [11]. A CDN is composed of a variety of points of presence which are essentially proxies containing the most popular content offered on the CDN [2] [13]. The core aspect of CDN is the hierarchical design in which top-level, origin servers contain all available content and lower-level nodes contain only the most frequently requested content in a the last mile, i.e., close to customers in a single geographic region [3].

Similarly to other large-scale enterprise systems, the daily amount of log lines produced by CDNs is measured in the tens or hundreds of millions or maybe even billions. As the manual analysis of such datasets is impractical and often impossible, it is a viable approach to rely on machine learning algorithms which automatically process log lines and discover interesting patterns. One such ML-based approach is error log clustering [5].

Neha et al [4]used a web log analyzer tool called Web Log Expert to identify users' behavior in terms of number of hits, page views visitors and bandwidth. An in-depth analysis of user behaviour of the NASA website was performed in

[15]. The goal of that research was to obtain information about top errors, websites and potential visitors of the site. The study showed that machine learning techniques such as association, clustering, and classification can be used to identify regular users of a website. The goal of study [7] was to determine the best approach for identifying user behavior, whether quantitative or qualitative methods are used. The researchers claimed that clustering users faces two challenges: reasoning and surrounding behavior. They came to the conclusion that combining qualitative and quantitative methodologies is the best way to understand user behavior. Haifei Xiang in [18] used weblog data to cluster user behavior and analyze user access patterns. The study explained how to analyze user behavior from weblog data and apply the K-means clustering algorithm. It considers the disadvantages of K-means to obtain local optimum solutions. Methods such as selecting the initial centers based on data sparsity were proposed, which may effectively minimize algorithm iteration time and increase clustering quality. In [16] the researchers clustered users into networks according to their browsing behaviour.Their methodology consisted of web log pre-processing and clustering with the K-means algorithm with the ultimate goal to group users into different categories and analyze their behaviour based on the category of the web sites with which they interact.

Log parsing is a technique for converting unstructured content from log messages into a format appropriate for data mining. The aim of the study [16] was to analyze how log parsing strategies using natural language processing affected log mining performance. The researchers utilized two datasets: the first consisted of log data collected in an aviation system which included over 4,500,000 messages gathered over the period of a year. The second dataset was log data from public benchmarks acquired from a Hadoop distributed file system (HDFS) cluster.

A novel way for presenting textual clustering to automatically find the syntactic structures of log messages logs collected on super-computing systems was proposed in [6]. The researchers managed to utilize their approach to extract meaningful structural and temporal message patterns. The goal of study [14] was to investigate user sessions via a frequent pattern mining approach in web logs.

3 Data exploration

The CDN service provider delivered logs in multiple batches, incrementally improving the data collection and anonymization methods for better fitting to the research goals. This work builds on multiple experimental data sets extracted from the base data set, which contains more than 2.2 billion CDN proxy log entries collected during a 7-day period. This dataset has 29 different features and most of them are categorical data - table 1 contains the list of features in the dataset. In the experimentation and prototyping phase of this research work we extracted different sample datasets from the logs.

Web and web proxy servers send a set of known HTTP status codes when processing a client request results in an error. These errors can be grouped

into client (HTTP status codes 4XX) and server errors (HTTP status 5XX). HTTP response status codes show whether or not a particular HTTP request was completed successfully. Responses can be grouped into the following five groups: (1) information response (100-199), (2) successful answers (200-299), (3) redirects (300-399), (4) client errors (400-499), and (5) server errors (500-599).

We considered a log line an error if the HTTP status code logged was equal or greater than 400. We started with the assumption that client side errors if the status code starts with 4 and it will be considered as server side error if the status code starts with 5.

Feature	Description
statuscode	HTTP response status codes
contenttype	Indicates the media type of the resource
protocol	Http version
contentlength	The size of the resource, in decimal number of bytes
timefirstbyte	time from request processing until the first byte
timetoserv	time needed to process the request
osfamily	Client's device Operating System
sid	id uniq to a streaming session
cachecontrol	Directives for caching mechanisms in both requests and responses
uamajor	User-Agent'version, eg. browser's version
uafamily	User-Agent,usually client's app
devicefamily	Type of the device
fragment	video or file fragment identifier
path	URL (address of request)
timestamp	Arrival time of the request
contentpackage	VoD asset identifier
coordinates	Long. and lat. of the client based on geoip lookup
livechannel	Live TV channel name
devicemodel	Client's device model
devicebrand	Client's device brand
host	Specifies the domain name of the server (for virtual hosting)
method	Http request method eg. Get,post
manifest	resource which the browser should cache for offline access
assetnumber	VoD asset encoding version
hit	HTTP request was a cache hit or miss
cachename	cache's hostname
popname	cache's location
uid	id unique to a single user

Table 1. List of log line features

Apart from the HTTP status codes, the log lines contained the following groups of features: (1) user device details (e.g., operating system, device brand, user agent), (2) HTTP request-related information (e.g., was the request a cache hit or not, protocol version, content type and path), (3) CDN node information

(host, cache, point of presence - PoP), as well as (4) customer-specific information (geo coordinates, user identifier, session identifier).

`Figure` 1 presents a subset the CDN log dataset with a subset of key features.

1	statuscode	contenttype	protocol	contentlength	timefirstbyte	timetoserv	maxage	osfamily	sid	
2	404		HTTP/1.1	0	0.069129	0.069188		0	3815	
3	404		HTTP/1.1	0	0.077291	0.077333		0	3320	
4	404	application/o	HTTP/1.1	0.206349206	0.049394	0.04945	3600	0	4170	
5	404		HTTP/1.1	0	0.06906	0.069112		3	4518	
6	404		HTTP/1.1	0	0.104557	0.1046		0	4727	
7	404	application/o	HTTP/1.1	0.206349206	0.028967	0.029026	3600	0	5682	
8	404	text/html; cha	HTTP/1.1	0.285714286	0.037654	0.037712		1	14102	
9	404	text/html; cha	HTTP/1.1	0.285714286	0.037481	0.037548		1	16121	
10	404	text/html; cha	HTTP/1.1	0.285714286	0.03824	0.038302		1	16329	
11	404	application/o	HTTP/1.1	0.206349206	0.028619	0.028683	3600	0	17075	
12	404	text/html; cha	HTTP/1.1	0.285714286	0.037611	0.037676		1	17431	

Fig. 1. Sample CDN logs data

4 Feature selection methodology and results

The data dimensionality problem was solved by utilizing a feature selection technique to reduce the number of features before any experiments. Our goal in this step was to identify the most important features related to the HTTP status codes created by the CDN hosts. For this purpose we used the following four feature selection techniques:

- ChiSquare: It works on testing whether a target value depends on input variable or not, all features will be scored and the higher scores indicate higher dependency on the target value [8].
- Correlation-based Feature selection: in this technique the linear relationship between variables is analysed and the results indicate the similarity of variables [12].
- ExtraTreeClassifer: This technique classifies the features using a collection of decision trees from a forest. A random sample of features will be extracted from a set of features and in every round the best features will be selected by each decision tree[10].
- Forward Feature Selection: In this technique an evaluation function will be used and in each iteration one feature will be added to the model then the results will be evaluated. The added feature will be kept in the output set of features if there was an improvement in the results. The process is repeated until the model no longer has any (new) improvements [17].

The goal of the feature selection phase was to reduce the number of attributes in a log record to a minimal representative subset. The one-hot encoding technique was used to transform the categorical features into numerical values - see `Figure` 2 which visualizes a subset of the resulting dataset.

	protocol_HTTP/1.1	devicebrand_0.0	devicebrand_1.0	devicebrand_15.0	devicebrand_2.0	devicebrand_3.0	devicebrand_4.0	devicebrand_5.0	devicebrand_6.0
0	1	1	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0	1
2	1	1	0	0	0	0	0	0	0
3	1	0	1	0	0	0	0	0	0
4	1	1	0	0	0	0	0	0	0
...	...	...	...	...	...	...	...	...	...
13860	1	1	0	0	0	0	0	0	0
13861	1	1	0	0	0	0	0	0	0
13862	1	1	0	0	0	0	0	0	0
13863	1	1	0	0	0	0	0	0	0
13864	1	1	0	0	0	0	0	0	0

13865 rows × 211 columns

Fig. 2. sample converted categorical data to numerical

4.1 Host-based feature selection

Each CDN host is responsible for a specific type of data, therefore we might be able to have a better representative of different requests in each host and that can lead us to the source of errors. In order to find the best features for host clustering we applied all techniques listed in the previous section for logs collected on each separate CDN host. Customers can use different services offered by a CDN service provider such as: web content, live TV, video on demand (VOD) and file caching. We chose three hosts with different services, considering the highest number of client error records in each. Our statistical analysis showed that for the live TV service (hosts 1 and 7) and for website service (host 3) had the highest number of error records. Our host-based feature selection showed that the relevant features were the same for all hosts as shown in Table 2. Note that we decided to focus our analysis on exactly those CDN hosts which produced the most error log lines and ignored the other CDN hosts whose logs were present in the dataset.

Selected Features for Host 1,3 and 7
statuscode, protocol, contentlength, timefirstbyte timetoserv, osfamily, uamajor, uafamily, devicefamily,path, device brand, method, Live channel

Table 2. Selected features for hosts

5 Results

The goal of the clustering step was to identify a set of error prototypes. Having these prototypes helps focusing on services and parts of CDN infrastructure which produce the most errors.

In this study we used *K-means*, and *K-modes* algorithms [3]. K-modes is one of the most well-known techniques for clustering categorical data [9]. Instead of utilizing the default parameter for K-means [4] and K-modes [5], we selected the optimal hyper-parameters with grid search. The values of these parameters are shown in tables 3 and 4 below.

Parameters	**Values**
`n_clusters`	6
`init`	k-means++
`max_iter`	300
`n_init`	10
`random_state`	0

Table 3. Parameters used for K-Means Clustering

Parameters	**Values**
`n_clusters`	8
`init`	Huang
`n_init`	5
`verbose`	1

Table 4. Parameters used for Kmodes Clustering

5.1 Host-specific error clustering

Hosts correspond to distinct services offered by the CDN network, e.g., video on demand, live TV or file services. We filtered the dataset and extracted all log lines with `statuscodes` ≥ 400.

In the dataset we identified log records from 57 hosts, but only 20 were actually valid hosts, the rest were created for different processes such as testing. After analyzing the valid hosts we found out that most of them only have a small number of records which can be easily analysed without clustering. Accordingly we only selected those hosts that had more than 1000 data points. As a result we ended up having three valid hosts 1 (live TV), 5 (website) and 7 (live TV). At this point we created three different sample data sets which only include the records from each specific host. Next we analysed them with different clustering algorithms.

[3] K-means can be used with categorical data after one-hot encoding.
[4] https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html
[5] https://pypi.org/project/kmodes/

For these experiments we used the features selected from Section 4.1, then we applied the K-Means and K-modes clustering algorithms.

5.2 Host 1 - Live TV

In our first experiment we used K-means with one hot encoding of categorical features [6]. The total number of data points for Host 1 was 13886 which were grouped into 6 different clusters. Most of the errors in this host were server errors, with a smaller number of client errors present.

Table 5 and figure 3 show the error log clusters for host 1.

K-means One hot encoding for host 1			
cluster	datapoints	status code	Relationship with other attributes
0	4676	412 (5 records) 500 (37 records) 502 (4634 records)	Protocol =HTTP 1.1, device brand =0,2,3,4,5,8 , Method =GET, device family=1,4,6,10,13,24,38,49,51,58,66,36,650, uafamily=1,5, Live channel =many , osfamily =1
1	1523	403 (12 records) 502 (1510 records) 503 (1 records)	Protocol =Http 1.1 , device brand =nan, method =GET , device family =0 ,nan, uafamily=1,3,6,7,16,24,nan , live channel =many, os family=0,2,3,4,nan
2	5508	412 (15 records) 500 (82 records) 502 (5411 records)	Protocol =Http 1.1, device brand =0, device family=1, ua family =2, live channel=many, os family= 1
3	753	500 (17 records) 502 (736)	Protocol =Http1.1, device brand= 2,5,6, method =GET, device family =many, ua family= 1,2, live channel =many, os family = 1
4	389	500 (1 records) 502 (388 records)	Protocol =Http 1.1, device brand = 4, method= GET, device family =8, ua family = 8, live channel = many, os family = 6
5	1016	403 (9 records) 416 (26 records) 502 (981 records)	Protocol =Http 1.1, device brand =1, method=GET, device family =2,5,7,9,11, ua family= 1,6,8,10, live channel= many, os family=3,5,7

Table 5. Host 1 clustering using K-Means

We clustered the error logs collected for host 1 with the K-Mode algorithm and with one hot encoding. The results were very similar to the previous experiment, one difference was that here we had 8 clusters. As can be seen in `Figure` 3 this host had a significant number of server errors 502 which shows that it most probably suffered from a traffic overload. Further analyses is warranted to investigate the temporal aspect of that overload.

[6] https://towardsdatascience.com/what-is-one-hot-encoding-and-how-to-use-pandas-get-dummies-function-922eb9bd4970

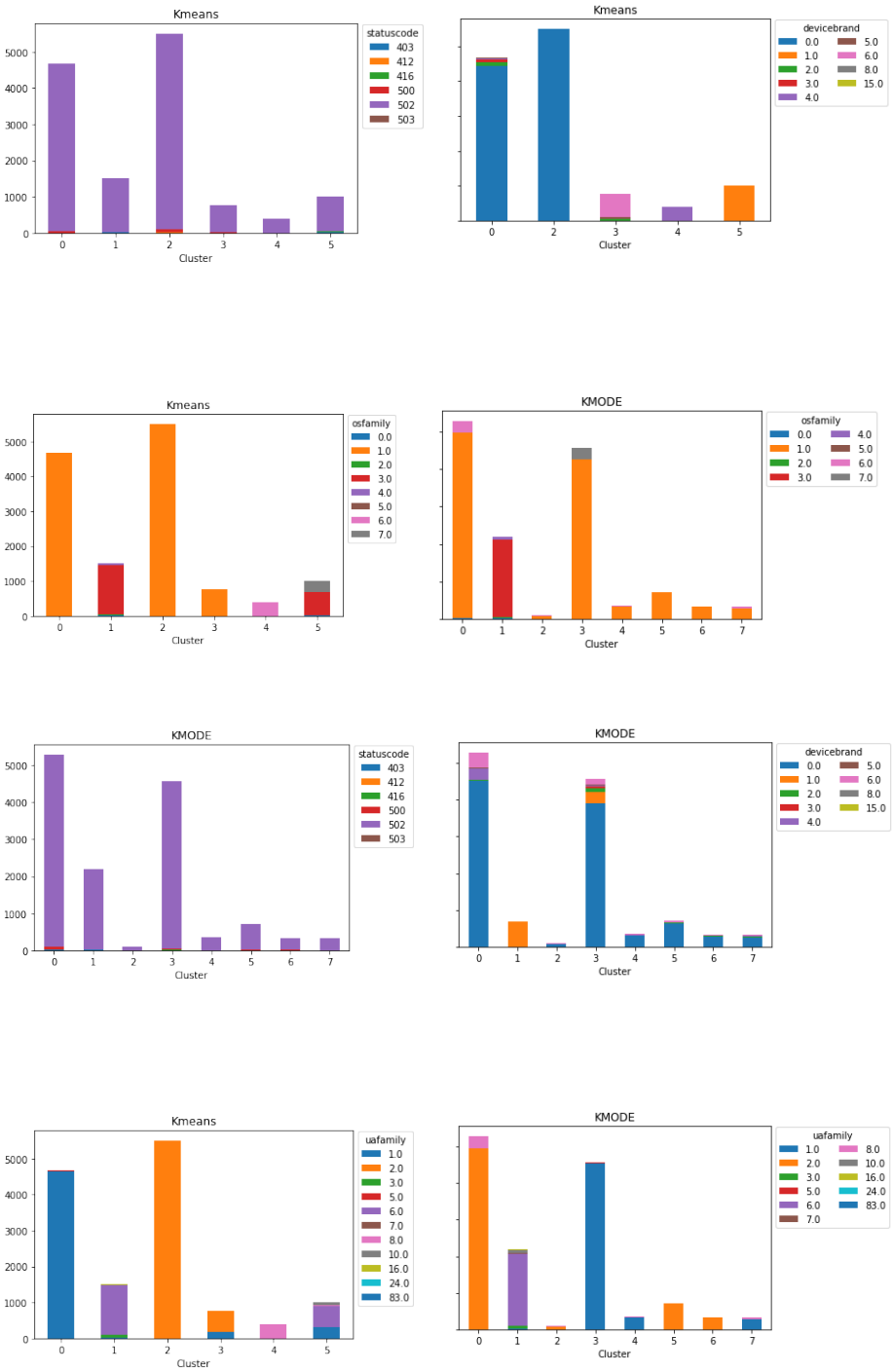
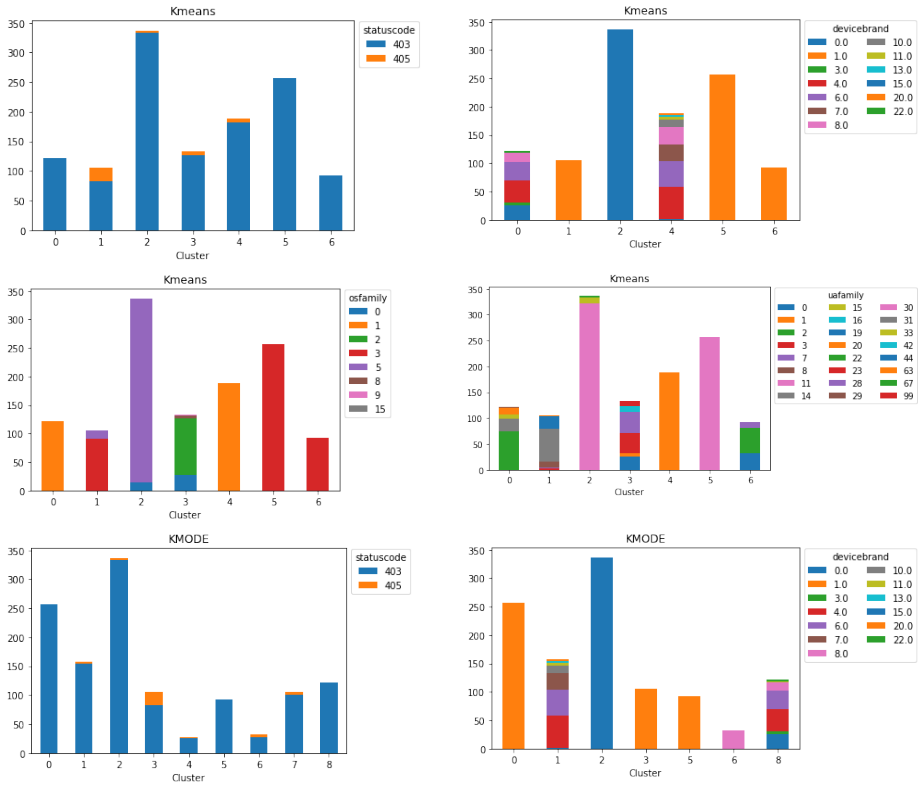


Fig. 3. Host 1 -Cluster using K-means and K-mode

5.3 Host 3 - Web content

There were 1235 errors logged on Host 3, 1197 forbidden requests (HTTP `statuscode` 403) and 40 method not allowed errors (HTTP `statuscode` 405). A total of 7 clusters were identified with k-means and 9 clusters with K-modes.

These errors corresponded to customers using different `devicefamily`, `devicebrand`, `osfamily` and `uafamily` devices. The feature distribution in each cluster is shown in `Figure` 4. They were most probably caused by customer device bugs or malicious activity. Note that clusters 3, 4 and 7 are not shown in the bar charts in `Figure` 4 due to a small number of the data points.



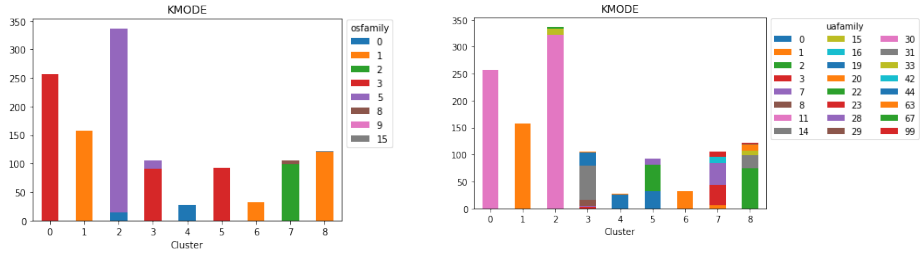


Fig. 4. Host 3 -Cluster using K-means and K-mode

5.4 Host7 - Live TV

Host 7 logged 31492 error records. 20272 bad requests (HTTP status code 400), 11078 precondition failed (HTTP 412), 129 service unavailable (HTTP 503) and 12 bad requests (HTTP 403). Our clustering approach grouped these records into 7 clusters.

In this host we noticed that the number of errors was significantly larger compared to Host 1 and 3 and all records corresponded to the same path with the same `device family, uafamily` and `osfamily`. After investigating we found out that these are actually a result of crawling. `Figure` 5 is an illustration of different features in each clusters.

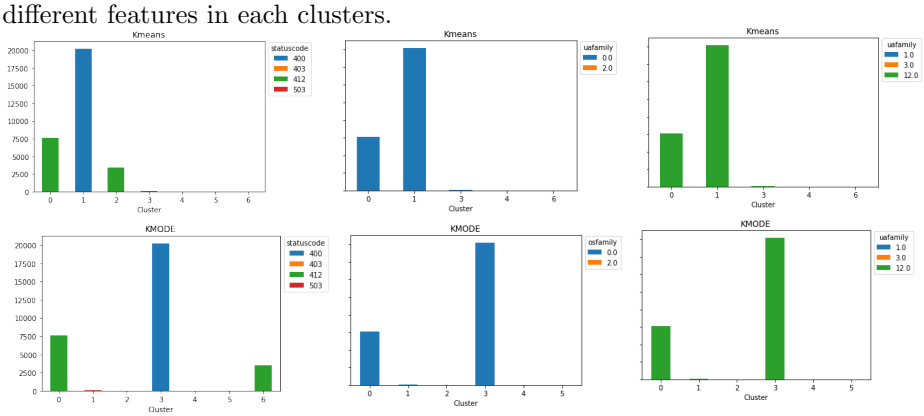


Fig. 5. Host 7 -Cluster using K-means and K-mode

5.5 Discussion

Error log clustering in content delivery networks is a less-frequently analysed topic, although it might allow CDN operators to pinpoint frequent error clusters and allow them to mitigate their causes. In that context, the log analysis approach prestented in this paper provides valuable novel information to CDN operators. We found that the CDN hosts analyzed in our experiments tended to produce different sets of errors. Host 3, a web server, only logged client errors 403 and 405, which might result from bugs in customer devices/browsers. On live

TV host 1 the vast majority of errors were produced by the CDN nodes (HTTP status code ≥ 500) which might point to a software issues or an overload on that specific host.

We were not completely satisfied with the results of our clustering approach, as ideally we would like to see clusters that clearly separate the different kinds of errors closest to the error prototypes (i.e., cluster centroids). We found that some of the identified clusters contained multiple HTTP error codes. This does not necessary mean that the clustering is bad. When we examined the sessions at the different hosts, we discovered that even the very same sessions can produce multiple error codes. Another, simpler explanation might be, that same type of resources just cause different errors on different resources, as server errors when something goes wrong within the CDN infrastructure, or client errors when someone simply provides a non existing URL. These explanations however naturally require further investigations.

6 Conclusion and Future work

The goal of this research was to apply different clustering techniques for mining value-added information from real-life CDN logs. We worked with a dataset consisting from 2.2 billion log lines collected over a 1-week period on multiple CDN hosts offering video on demand, web content, file and live TV services.

In the data exploration phase we filtered out the log lines corresponding to HTTP error codes (≥ 400). We performed feature selection with different techniques and thereby reduced the total number of features from 29 to 10. Additionally, we decided to separately analyze the error logs corresponding to hosts offering video on demand, live TV and web content. We experimented with the K-means and K-mode algorithms and identified meaningful and explainable error clusters, which can be utilized by CDN operators to obtain increased situational awareness and optimize the workings of their systems. We also introduced a novel, visual representation method (figures 3, 4 and 5) which visually conveys information about clusters of observations described by multiple categorical features.

We analyzed all error logs collected over a period of a single week on each of the CDN hosts, which means that our analysis was time-agnostic. In future we plan to incorporate the temporal aspect into our analysis. Additionally, we also plan to perform user session-level error analysis and thereby better understand why some communication sessions end in (HTTP) errors.

References

1. Jakub Breier and Jana Branišová. Anomaly detection from log files using data mining techniques. In *Information Science and Applications*, pages 449–457. Springer, 2015.
2. Luca Brilli, Romano Fantacci, Tommaso Pecorella, and Tiziano Ionta. The effect of video caching on network resource planning—a real-case study. In *In 2016 AEIT International Annual Conference (AEIT), IEEE*, pages 1–6, 2016.

3. Mirosław Czyrnek, Ewa Kuśmierek, Cezary Mazurek, Maciej Stroiński, and Jan Węglarz. Cdn for live and on-demand video services over ip. In *Content Delivery Networks*, pages 317–342. Springer, 2008.
4. Neha Goel and CK Jha. Analyzing users behavior from web access logs using automated log analyzer tool. *International Journal of Computer Applications*, 62(2), 2013.
5. LK Grace, V Maheswari, and Dhinaharan Nagamalai. Analysis of web logs and web user in web mining. *arXiv preprint arXiv:1101.5668*, 2011.
6. Sourabh Jain, Inderpreet Singh, Abhishek Chandra, Zhi-Li Zhang, and Greg Bronevetsky. Extracting the textual and temporal structure of supercomputing logs. In *2009 International Conference on High Performance Computing (HiPC)*, pages 254–263. IEEE, 2009.
7. Andreas Johansson. Clustering user-behavior in a collaborative online social network: A case study on quantitative user-behavior classification, 2016.
8. Samina Khalid, Tehmina Khalil, and Shamila Nasreen. A survey of feature selection and feature extraction techniques in machine learning. In *2014 science and information conference*, pages 372–378. IEEE, 2014.
9. Mingshuang Li, Yihui Zhou, Wenru Tang, and LaiFeng Lu. K-modes based categorical data clustering algorithms satisfying differential privacy. In *2020 International Conference on Networking and Network Applications (NaNA)*, pages 86–91. IEEE, 2020.
10. Solichin Mochammad, Young-Jin Kang, Yoojeong Noh, Sunhwa Park, and Byeongha Ahn. Stable hybrid feature selection method for compressor fault diagnosis. *IEEE Access*, 9:97415–97429, 2021.
11. Erik Nygren, Ramesh K Sitaraman, and Jennifer Sun. The akamai network: a platform for high-performance internet applications. *ACM SIGOPS Operating Systems Review*, 44(3):2–19, 2010.
12. Nachirat Rachburee and Wattana Punlumjeak. A comparison of feature selection approach between greedy, ig-ratio, chi-square, and mrmr in educational mining. In *2015 7th International Conference on Information Technology and Electrical Engineering (ICITEE)*, pages 420–424. IEEE, 2015.
13. Moisés Rodrigues, André Moreira, Ernani Azevedo, Marcio Neves, Djamel Sadok, Arthur Callado, Josilene Moreira, and Victor Souza. On learning how to plan content delivery networks. In *Proceedings of the 46th Annual Simulation Symposium*, pages 1–8, 2013.
14. Sheetal Sahu, Praneet Saurabh, and Sandeep Rai. An enhancement in clustering for sequential pattern mining through neural algorithm using web logs. In *2014 International Conference on Computational Intelligence and Communication Networks*, pages 758–764. IEEE, 2014.
15. KR Suneetha and Raghuraman Krishnamoorthi. Identifying user behavior by analyzing web server access log file. *IJCSNS International Journal of Computer Science and Network Security*, 9(4):327–332, 2009.
16. Michal Turčaník. Web users clustering by their behaviour on the network. In *2020 New Trends in Signal Processing (NTSP)*, pages 1–5. IEEE, 2020.
17. Yap Bee Wah, Nurain Ibrahim, Hamzah Abdul Hamid, Shuzlina Abdul-Rahman, and Simon Fong. Feature selection methods: Case of filter and wrapper approaches for maximising classification accuracy. *Pertanika Journal of Science & Technology*, 26(1), 2018.
18. Haifei Xiang. Research on clustering algorithm based on web log mining. In *Journal of Physics: Conference Series*, volume 1607, page 012102. IOP Publishing, 2020.

Authors Index