Transactions on Machine Learning
and Data Mining
Vol. 7, No. 1 (2014) 3-25
ISSN 1864-9734
ISBN: 879-3-942952-28-6

ibai Publishing
www.ibai-publishing.org

Experiencing the Shotgun Distance for Time Series Analysis

Patrick Schäfer

Zuse Institute Berlin, Berlin, Germany
patrick.schaefer@zib.de

Abstract. Similarity search is a core functionality in many data mining algorithms. Over the past decade algorithms were designed to mostly work with human assistance to extract characteristic, aligned patterns of equal length and scaling. We propose the *shotgun distance* similarity measure that extracts, scales, and aligns segments from a query to a sample time series. This greatly simplifies the time series analysis task of those time series produced by sensors. We show the applicability of our shotgun distance in the context of hierarchical clustering of heraldic shields, and human motion detection. A time series is segmented using varying lengths as part of our *shotgun ensemble classifier*. This classifier improves the best published accuracies on case studies in the context of bioacoustics, human motion detection, spectrographs or personalized medicine. Finally, it performs better than state of the art on the official UCR classification benchmarks.

Keywords: Time Series, Distance Measure, Similarity, Classification, Hierarchical Clustering, Shotgun Analysis, Segments

1 Introduction

Time series result from recording data over time. The task of analyzing time series data [1–3] is difficult as the data may be recorded at variable lengths, and are erroneous, extraneous due to noise, dropouts, subtle distinctions and highly redundant due to repetitive (sub-)structures. Application areas include ECG [4] or EEG signals, human walking motions [5], or insect wing beats [6], for example.

Empirical evaluation suggests that distance measures like the Euclidean distance (ED) or dynamic time warping (DTW) are hard to beat [1, 2]. However, these have some known shortcomings. The ED does not provide horizontal alignment or support variable length time series. DTW provides warping invariance which is a peak-to-peak

and valley-to-valley alignment of two time series, which fails if there is a variable number of peaks and valleys.

Figure 1 shows a hierarchical clustering of a synthetic dataset. It consists of three types of shapes, which have variable lengths and phase shifts. Even though the dataset is very simple, the distinguishing powers of both the ED and DTW distance measures are very disappointing. The ED fails to separate the shapes as it neither supports horizontal alignment nor variable lengths. Neither does DTW result in a satisfying clustering as it fails to separate the triangles from the sine waves. Our shotgun distance clusters all shapes correctly. This toy example illustrates some of the difficulties resulting from time series similarity. In general, several sources of invariance like *amplitude/offset,*

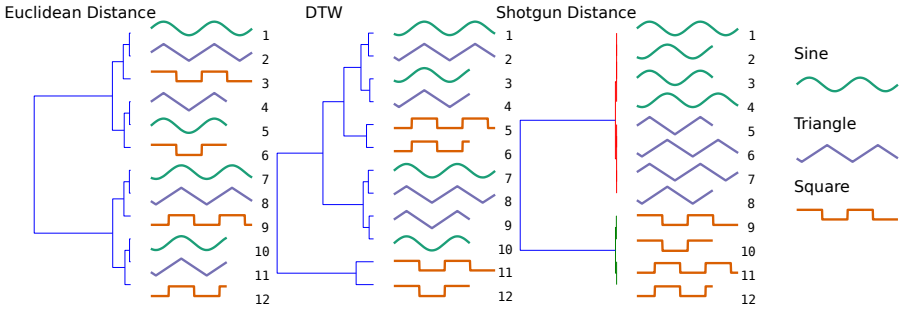


Fig. 1: A hierarchical clustering of a synthetic dataset based on three similarity measures. There are three types of curves: sine, square, triangle.

warping, phase, uniform scaling, occlusion, and complexity have been identified [7]. Both, ED and DTW calculate the distance between two entire time series to determine their similarity. To make these applicable a significant amount of time and effort has to be spent by a domain expert to filter the data and extract equal-length, equal-scale, and aligned patterns. In our toy example the shapes have to be aligned and trimmed to equal-length for the ED and DTW distance measures to give meaningful results. Human assistance significantly eases the subsequent data mining task both in terms of the cost of the execution time and the complexity of the algorithm. However, human assistance is often too time consuming and expensive [8, 9]. Only few algorithms exist that deal with the data 'as is'. These algorithms are based on matching time series by their structural similarity [10, 11]. The idea is to deliberately ignore some data, by extracting local, representative time segments from a time series. Ignoring the appropriate data is a non-trivial task. As traditional data mining algorithms are not easily applicable to raw datasets, international competitions were staged like identifying whale calls [12], human walking motions [12] and flying insects [6].

Our work introduces a simple and novel similarity measure for time series similarity search. *Shotgun distance* vertically and horizontally aligns time series segments (subsequences) of a query to a sample time series (Figure 2). Thereby it avoids preprocessing the data for alignment, scaling or length. This is achieved by breaking the query into disjoint subsequences of fixed length first. Next, each query subsequence is slid

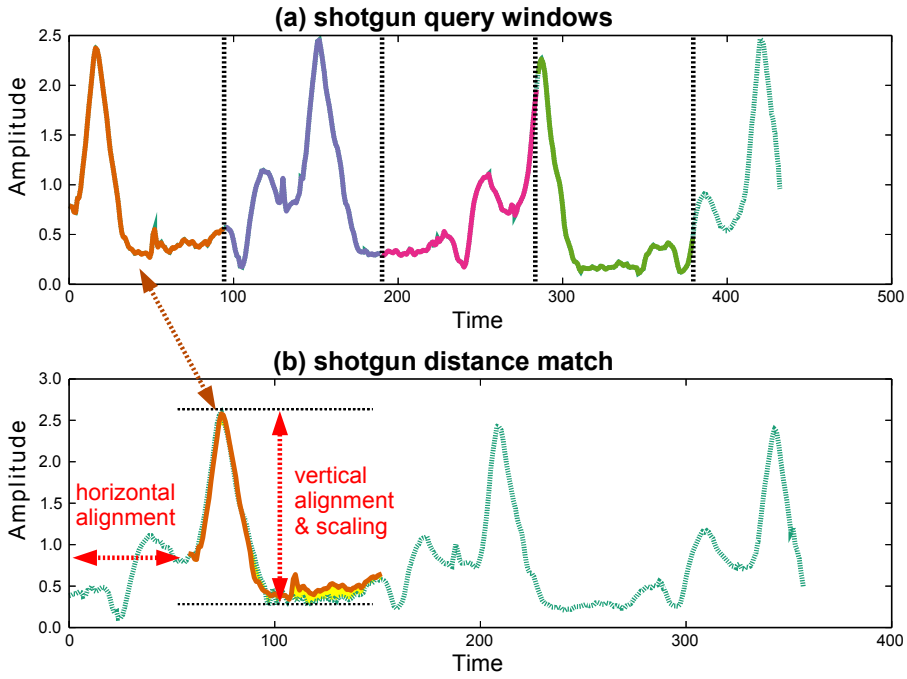


Fig. 2: Shotgun distance consists of segment extraction, horizontal and vertical alignment, and scaling.

along a time series sample to find the best matching position in terms of minimizing a distance measure (horizontal alignment). These distances are aggregated. The sample that minimizes this aggregated distance is the 1-nearest-neighbor (1-NN) to a query and most similar. Normalization is applied prior to each distance computation, to provide the same vertical alignment and scaling of each subsequence. Our contributions are as follows:

- Section 2 presents the motivation and related work on time series analysis.
- We introduce the shotgun distance that provides vertical scaling and horizontal alignment in Section 3.
- We present the shotgun ensemble classifier, which is an ensemble of 1-NN classifiers utilizing the shotgun distance at multiple subsequences lengths in Section 3.4.
- Two pruning strategies are presented which significantly reduce the computational complexity by one order of magnitude in Section 3.5.
- We present case studies for hierarchical clustering and classification in Section 4. Our classifier is significantly more accurate than state of the art on 5 case studies and the UCR benchmark datasets.

A preliminary version of this paper has been published in [13]. In addition this paper contains:

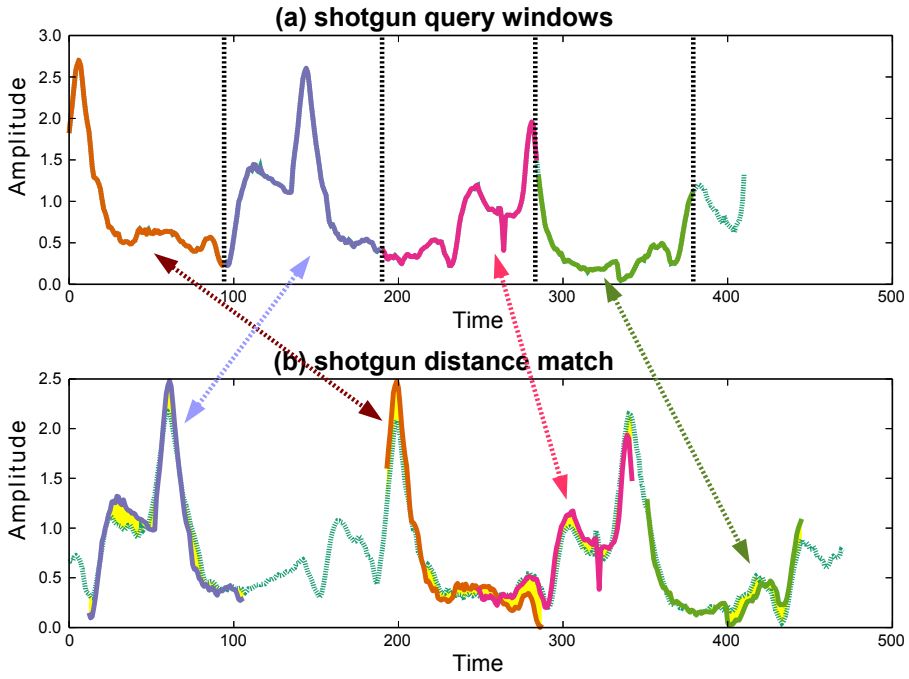


Fig. 3: Matching the gait cycles in the query to the sample is complicated due to different amplitudes, phase-shifts, variable lengths and noise.

- an extended description of the shotgun ensemble classifier,
- experiments using hierarchical clustering and model data,
- an analysis of the shotgun distance parameter space,
- new results for computational bioacoustics,
- a discussion on the impact of the design decisions.

2 Motivation & Related Work

The utility of the shotgun distance is tied to the observation that a multitude of signals are composed of characteristic patterns. Consider human walking motions [5] as a concrete example. The data was captured by recording the z-axis accelerometer values of either the right or the left toe. The difficulties in this dataset result from variable-length gait cycles, gait styles and pace due to different subjects throughout different activities. Figure 3 illustrates the walking motion of a subject, that is composed of 4 gait cycles. Classifying walking motions is difficult, as the samples are not preprocessed to have an approximate alignment, length, scale or number of gait cycles and are typically noisy.

Shotgun distance reduces the need for cost-ineffective preprocessing by vertically aligning and horizontally scaling the query to a sample time series. It is an analogy to *Shotgun Sequencing* [14], the process of breaking up a sequence into numerous small

segments which are resembled based on overlaps. Figure 3 (bottom) illustrates the result of this process. The distance between the 4 gait cycles in the query and the sample are minimized, even though these differ in scale, have a variable length, and a phase-shift and noise occur.

The quality of the shotgun distance is subject to two parameters (Figure 2):

1. *horizontal alignment* using the *window* length: an integer parameter which is limited by the length of the longest query.
2. *vertical alignment* using the *mean*: a Boolean parameter which defines if the mean should be subtracted prior to the distance calculations. The standard deviation is always normed to 1 to obtain the same scaling. Surprisingly, the mean normalization has not been considered to be a parameter before in literature.

The *window length* parameter controls the length of the segments and depends on the length of the characteristic patterns in the dataset. Furthermore, it regulates how much information on the ordering of the values within the time series is incorporated into the matching-process. For long window lengths the whole query will be treated as a single pattern. This mostly happens with signals which were preprocessed by a human for alignment and length. In contrast, human motions contain repetitive gait cycles. Aligning any gait cycle in the query to any gait cycle in the sample is equivalent. Thus, the ordering information is less relevant, resulting in a window length that should be roughly equal to one gait cycle.

2.1 Related Work

Time series similarity search is a complex task for a computer. It is non trivial to extract a general statistical model from time series as these may show varying statistical properties with time. Classical machine learning algorithms degenerate due to the high dimensionality of the time series and noise [15]. Approaches can be characterized by (a) they try to find a similarity measure that resembles our intuition of similarity in combination with 1-NN classification (*shape-based*) or (b) they transform the data into an alternative data space to make existing data mining algorithms applicable (*structure-based*) [1, 16, 17]. The UCR time series classification datasets [3] have been established for reference [1–3, 16, 11].

Shape-based techniques include 1-NN Euclidean Distance (ED), or 1-NN DTW [18, 19] and are used as the reference [2]. However, shape-based techniques fail to classify noisy or long data.

Structure-based techniques [1, 10, 16, 11, 20, 21] are based on data mining algorithms such as SVMs, decision trees, or random forests in combination with feature extraction. Feature extraction techniques include DFT [22], PLA [23], SFA [24], SAX [25], or shapelets. By transforming time series data into an alternative space (i.e. using functional data analysis) the performance of classifiers can be improved [1]. However, the authors failed to show a significant improvement over 1-NN DTW. Shapelet classifiers [11, 20, 21] extract representative variable-length subsequences (called *shapelets*). A decision tree is build using these shapelets within the nodes of the tree and distance threshold for branching. One algorithms deals with classification on raw data [10]. The

shotgun classifier is inspired by shotgun sequencing introduced to find an alignment of two DNA or protein sequences [14]. Shotgun sequencing was used to find the horizontal displacements of steel coils [26]. To find the horizontal displacement the authors use the median on the differences of the calculated starting positions for every pair of subsequences.

3 Shotgun Distance

3.1 Definitions

A time series consists of a sequence of real values:

$$T = (t_1, \ldots, t_n) \tag{1}$$

This time series is split into subsequences (time segments) using a windowing function.

Definition 1. *Windowing: A time series $T = (t_1, \ldots, t_n)$ of length n is split into fixed-length windows $S_w(a) = (t_a, \ldots, t_{a+w-1})$ with length w and offset a in T. Two consecutive windows can* overlap *within an interval of $[0, w)$. Given the overlap, there are $\frac{(n-w)}{(w-overlap)}$ windows in T:*

$$windows(T, w, overlap) = \bigcup_{i=0}^{\frac{(n-w)}{(w-overlap)}} S_w(i \cdot (w - overlap) + 1) \tag{2}$$

To *vertically* align two samples, the query window and the sample window are typically z-normalized by subtracting the mean and dividing by the standard deviation:

$$\hat{\omega}(T, w, overlap) = z\_norms(windows(T, w, overlap)) \tag{3}$$

However, the mean normalization is treated as a parameter of our model and can be enabled or disabled. For example, heart beats have to be compared using a common baseline but the pitch of a bird sound can be significant for the species. Commonly, the similarity of two time series is measured using a distance measure. The shotgun distance is a distance measure that minimizes the Euclidean distance between each disjoint window in the query Q and the sliding windows in a sample S. For example, each gait cycle is slid along a longer walking motion to find the best matching positions by minimizing the Euclidean distance.

Definition 2. *Shotgun distance: the shotgun distance $D_{shotgun}(Q, S)$ between a query Q and a sample S is given by aggregating the minimal Euclidean distance $D(Q_a, S_b)$ between each disjoint query window $Q_a \varepsilon \hat{\omega}(Q, w, 0)$ and each offset b in S, represented by the sliding windows $S_b \varepsilon \hat{\omega}(S, w, w-1)$:*

$$D_{shotgun}(Q, S) = \sum_{a=1}^{len(\hat{\omega}(Q,w,0))} \min \{D(Q_a, S_b) \mid S_b \varepsilon \hat{\omega}(S, w, w-1)\} \tag{4}$$

Algorithm 1 The shotgun distance.

```
Input:  query     : time series
        sample    : time series
        W_LEN     : the window length
        MEAN_NORM: boolean parameter to norm the mean
Output: The distance between query and sample

double ShotgunDistance(query, sample, W_LEN, MEAN_NORM)
(1) totalDist = 0.0
    // for each disjoint query window
(2) for q in disjoint_windows(query,W_LEN,MEAN_NORM)
(3)   qDist = MAX_VALUE
      // find the position that minimizes the distance
(4)   for s in sliding_windows(sample, W_LEN, MEAN_NORM)
(5)     qDist = min(qDist, EuclideanDist(q,s))
(6)   totalDist += qDist
(7) return totalDist
```

This definition resembles the extraction of characteristic patterns (i.e. the gait cycles), and the scaling and aligning of the patterns. The latter provides invariance to the time ordering of the patterns and allows for comparing variable length time series. The shotgun distance is equal to the Euclidean distance for n equal to w.

The shotgun distance is not a *distance metric* as it neither satisfies the symmetry condition nor the triangle inequality. This is a result of the use of disjoint query windows and sliding sample windows. As a consequence the shotgun distance does not allow for indexing (triangle inequality) and the nearest neighbor of X may not be the nearest neighbor of Y (symmetry).

3.2 Shotgun Distance Algorithm

The shotgun distance in Algorithm 1 makes use of the Euclidean distance, and can be tuned by the two parameters *window length* W_LEN and *mean normalization* MEAN_NORM (the standard deviation of q and s is always normed to 1 regardless of MEAN_NORM). It first splits the query into disjoint windows (line 2) and searches for the position in the sample that minimizes the Euclidean distance (line 4-5). Finally, the distances are accumulated for each query window (line 6).

Complexity: The computational complexity is quadratic in the length of the time series Q and S: for each query window, all sample windows are iterated and the Euclidean distance for each pair of windows is calculated. There are $\frac{|Q|}{w}$ disjoint query windows and $|S|-w+1$ sliding windows for window length w:

$$T(\text{Shotgun Distance}) = O\left(\underbrace{\frac{|Q|}{w}}_{\text{disjoint windows}} \cdot w \cdot \underbrace{(|S|-w+1)}_{\text{sliding windows}}\right) \tag{5}$$

$$\text{for } n = \max(|Q|,|S|) \Rightarrow O\left(n^2 - nw\right) \tag{6}$$

Algorithm 2 The Shotgun Classifier.

```
Input:  query     : time series
        samples   : a set of time series
        W_LEN     : the window length
        MEAN_NORM: boolean parameter
Output: The predicted label of the query

String predict(query, samples, W_LEN, MEAN_NORM)
(1) (dist, nn) = (MAX_VALUE, NULL)
(2) for sample in samples
(3)   D = ShotgunDistance(query, sample, W_LEN, MEAN_NORM)
(4)   if D < dist
(5)     (dist, nn) = (D, sample)
(6) return nn.label


Input:  samples   : a set of time series
        MEAN_NORM: boolean parameter
Output: a list of tuples [(accuracy, length)]

[(int,int)] fit(samples, MEAN_NORM)
(1) scores = []
    // search for best window lengths in parallel
(2) for len = maxLen down to minLen
(3)   correct = 0
(4)   for query in samples
(5)     nnLabel = predict(
            query,samples\{query},len,MEAN_NORM)
(6)     if (nnLabel==query.label) correct++
      // store scores for each window length
(7)   scores.push((correct, len))
(8) return scores
```

Note that for large window lengths $w \sim n$ this complexity is close to linear in n (like the Euclidean distance). For small window lengths $w \ll n$ the complexity is quadratic in n^2 (like DTW).

3.3 Shotgun Classifier

The shotgun classifier is based on 1-NN classification and the shotgun distance. Given a query, the *predict*-method (Algorithm 2) searches for the 1-NN to a query within the set of samples (line 3–5). Finally, the query is labeled by the class label of the 1-NN *nn*.

The *fit*-method (Algorithm 2) performs a grid-search over the parameter space using leave-one-out cross-validation (lines 4–8). It obtains the parameters that maximize the accuracy on the train samples. The accuracies for all window lengths starting from the *maxLen* (the length of the longest time series) down to *minLen* (line 7) are recorded. The *MEAN_NORM*-parameter is a Boolean parameter, which is constant for a whole dataset as opposed to setting it per sample or window.

3.4 Shotgun Ensemble Classifier

By intuition every dataset is composed of substructures at multiple window lengths caused by different walking motions, heart beats, duration of vocals, length of shapes.

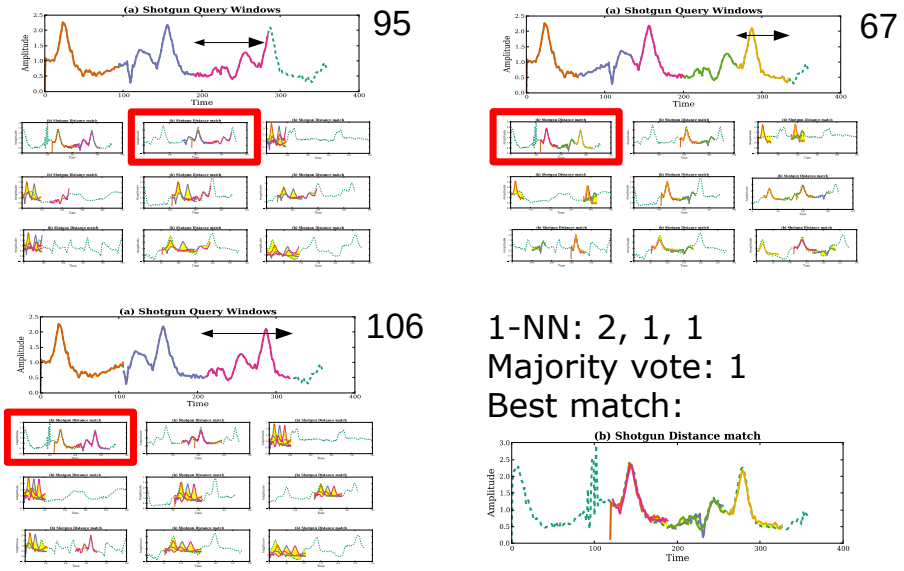


Fig. 4: The shotgun ensemble classifier using three window lengths: 67, 95 and 106. The 1-NN is searched for each query within the 9 samples. The matching positions are illustrated within each sample. This result in the nearest neighbors highlighted by the red rectangles: 2, 1, 1.

Algorithm 3 The shotgun ensemble classifier.

```
Input:  query     : time series
        samples   : a set of time series
        bestScore: the best accuracy
        windows   : a list of tuples [(accuracy, length)]
        MEAN_NORM: boolean parameter
Output: The predicted label of the query

String predictEnsemble(
    query, samples, bestScore, windows, MEAN_NORM)
    // stores for each window length a label
(1) windowLabels = []
    // determine the label for each window length
(2) for (correct, len) in windows
(3)   if (correct > bestScore*factor)
(4)     windowLabels[len] = predict(
            query, samples, len, MEAN_NORM)
(5) return most frequent label from windowLabels
```

For example, each human may have a different length of a gait cycle. To allow for datasets that contain patterns of different window lengths, we extend the shotgun classifier algorithm to support multiple window lengths, therewith making it an ensemble technique and further add to the robustness of the classifier. Figure 4 illustrate the shotgun ensemble classifier applied to walking motions using three different window lengths: 67, 95 and 106. The query is split into disjoint windows of the corresponding lengths and the 1-NN is searched within the 9 samples. Each window length results in its own 1-NN. A majority vote leads to the time series illustrated in the bottom right.

Algorithm 2 describes the shotgun classifier for one fixed window length. The ensemble classifier is defined in Algorithm 3, which allows for different substructural sizes within the time series. The fit-method in Algorithm 2 returns a list of scores, each one resulting from a different window length on the train samples. The shotgun ensemble classifier (Algorithm 3) classifies a query using the best window lengths from the list of scores. The best accuracy on the train samples is given by *bestScore.* Using a constant parameter $factor \varepsilon (0,1]$ and this *bestScore*, the optimal window lengths are given by:

$$correct > bestScore \cdot factor$$

For each of these window lengths the label is recorded (line 4). Finally, the most frequent class label is chosen from these labels (line 5). While it might seem that we add yet another parameter *factor*, the training of the shotgun ensemble classifier depends solely on the *factor* and *mean* parameters. The shotgun ensemble classifier model is derived from these two parameters using the *fit*-method, which returns the list of window scores. These scores are used as the model and to predict the label of an unlabeled query. In our experiments factors in between 0.92 to 1.0 were best throughout most datasets.

3.5 Pruning the Search Space

The rationale of search space pruning is to early abandon computations, as soon as these can not result in finding a new optimum. Previous work aims at stopping Euclidean distance calculations when the current distance exceeds the best distance found so far [11, 20, 21].

Early Abandoning: The purpose of the *ShotgunDistance*-method (Algorithm 4) is to accumulate the Euclidean distances for each query window. The Euclidean distance computations are pruned by reusing the best result *qDist* of the previous calculations (line 5). The *ShotgunDistance*-method is executed multiple times for each pair of query and sample. Passing the distance to the current calculation as *bestDist* allows for pruning calculations as soon as this *bestDist* is exceeded (line 7). Otherwise the sample is a new nearest-neighbor candidate and the distance is used to prune subsequent calls to *ShotgunDistance*. In the best case scenario, we have to compute the distance between one pair of time series and all other distance computations stop after one iteration of the for-loop in line 7.

Algorithm 4 Pruning techniques based on early abandoning.

```
double EuclideanDist(query, sample, bestDist)
(1) for i = 1 to len(query)
(2)   dist += (sample[i] - query[i])^2
      // early abandoning
(3)   if (dist > bestDist) return MAX_VALUE
(4) return dist

double ShotgunDistance(query, sample, W_LEN, MEAN_NORM, bestDist)
(1) totalDist = 0
(2) for q in disjoint_windows(query, W_LEN, MEAN_NORM)
(3)   qDist = MAX_VALUE
(4)   for s in sliding_windows(sample, W_LEN, MEAN_NORM)
        // early abandoning
(5)     qDist = min(qDist, EuclideanDist(q,s,min(qDist, bestDist)))
(6)   totalDist += qDist
      // early abandoning
(7)   if (totalDist > bestDist) return MAX_VALUE
(8) return totalDist

String predict(q, samples, W_LEN, MEAN_NORM)
[...]
(2) for sample in samples
(3)   D = min(D, ShotgunDistance(q, sample, W_LEN, MEAN_NORM, D))
[...]
```

Algorithm 5 Use an upper bound on the current accuracy.

```
[(int,int)] fit(samples, MEAN_NORM)
(1) scores = [], bestCorrect = 0
(2) for len = maxLen down to minLen
(3)   correct = 0
(4)   for q in [1..len(samples)]
(5)     nnLabel = predict(
          samples[q],samples\{samples[q]}, len, MEAN_NORM)
(6)     if (nnLabel==samples[q].label) correct++
(7)     if (correct+(len(samples)-q)) < bestCorrect*factor
(8)       break
(9)   bestCorrect = max(bestCorrect,correct)
[...]
```

Upper Bound on Accuracy: While lower bounding on distance computations aims at reducing the complexity in the length n, we present a novel optimization that also aims at reducing the complexity in the number of samples N. For each window length, the best achievable accuracy at any point is given by:

$$\text{correct} \leq (\text{current correct} + \text{remaining samples}) = N \tag{7}$$

Thus, we do not need to obtain the exact accuracy for a window length in Algorithm 5 (lines 7–8), if the remaining samples will not result in finding a better accuracy (or at least within *factor* to the best accuracy).

4 Experimental Evaluation

The utility of the shotgun distance is underlined by case studies and the UCR time series classification benchmark datasets [3]. Each dataset is split into two subsets: *train*

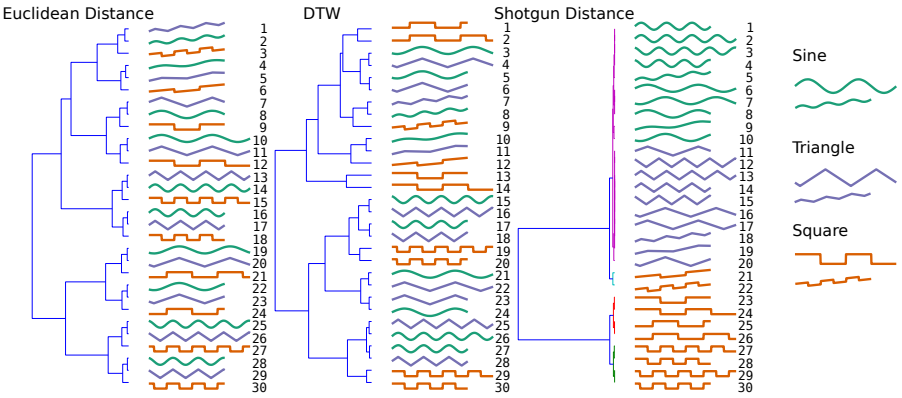


Fig. 5: Hierarchical clustering of a synthetic dataset based on three similarity measures. There are three types of curves: sine, square, triangle.

and *test*. By the use of the same train/test splits the results are comparable those previously published [1, 7, 2, 11, 20]. In all experiments we optimized the parameters of the classifiers based on the train dataset. The optimal set of parameters is then used on the test dataset. Our web page [27] contains a spreadsheet with all raw numbers and source codes. All benchmarks were performed on a shared memory machine running Linux with 8 Quad-Core AMD Opteron 8358 SE and Java JDK x64 1.7.

4.1 Hierarchical Clustering

In [7] five kinds of invariances for distance measures were presented:

1. *Amplitude/offset* invariance resulting from different scales like Celsius or Fahrenheit.
2. *Local scaling (warping)* invariance resulting from local distortions of a signal.
3. *Uniform scaling* invariance resulting from global distortions of a signal.
4. *Phase* invariance resulting from horizontal misalignment due to phase-shifts of periodic signals.
5. *Occlusion* invariance resulting from missing data.

The shotgun distance accounts for amplitude, local scaling, phase, local scaling and occlusion invariances. This is underlined by the following case studies. Unfortunately, good clustering results do not imply good classification results and vice versa, as the symmetry condition is not satisfied by the shotgun distance.

Hierarchical Clustering of Synthetic Data: We use a synthetic dataset to illustrate the utility of the shotgun distance in comparison to the Euclidean Distance (ED) and Dynamic Time Warping (DTW). The data consists of three types of shapes: *square, triangle, sine waves*. Each shape accounts for a separate class. The generated data has phase shifts, variable lengths, variable frequencies and a rising trend. A similarity measure needs to provide amplitude/offset, warping and phase invariance to separate the data

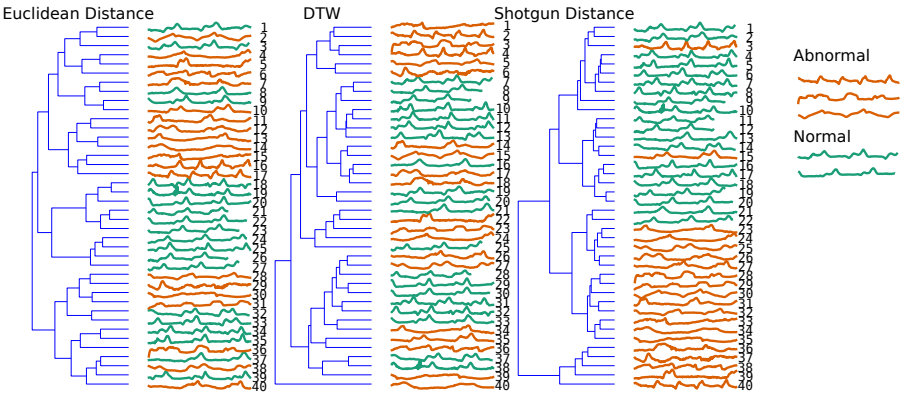


Fig. 6: Hierarchical clustering of a walking motions. There are two types of walking motions: normal and abnormal walk.

successfully. Figure 5 shows a hierarchical clustering of these wave forms. The ED performs very poorly at clustering the data. It clusters data with the same trend (signals 1-6) or length (10-12, 13-15, 19-21, etc.) independent of the form of the shape. Even though DTW provides invariance to local scaling, its performance is marginally better than that of the ED. Wave forms are clustered based on the number of valleys and peeks rather than their shape (signals 3-4, 5-6, 15-16, etc.). This is a know limitation of DTW. In contrast, our shotgun distance successfully separates all types of shapes. Each is clustered within separate branches of the dendrogram.

Hierarchical Clustering of Walking Motions: Figure 6 shows a hierarchical clustering of walking motions [5]. Each motion was categorized by the labels normal walk (green) and abnormal walk (orange). The difficulties in this dataset result from variable length gait cycles, gait styles and paces due to different subjects throughout different activities including stops and turns. A normal walking motion consists of multiple repeated similar patterns. The ED fails to identify the abnormal walking styles, as these are not separated from the normal walking motions. DTW provides invariance to phase shifts by a peak-to-peak and valley-to-valley alignment of the time series. This still does not result in a satisfying clustering as the abnormal and normal walking patterns are mixed within the same branches of the hierarchical clustering. Out shotgun distance separates the normal walking motions from the abnormal walking motions much clearer with just the 3rd and 15th walking motion being out of place.

Hierarchical Clustering of Heraldic Shield: Figure 7 shows a hierarchical clustering of the shape of heraldic shields [28]. There shields come from three countries: Spanish (purple), Polish (orange), French (green). The shape of the shields differ greatly based on their origin. The method used to transform the shape of a shield to a time series is described in [28]. Other than the previous two case studies, these time series do not have any periodicity. The dataset requires warping and scaling invariances for successful clustering.

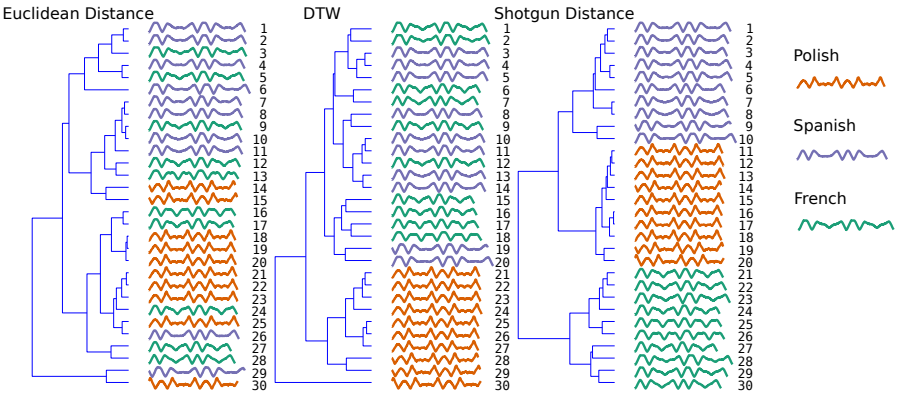


Fig. 7: Hierarchical clustering of a heraldic shield outlines. There are three types of shields: Spanish, Polish and French.

The ED clusters the time series based on their length and not based on their shape, which results in a visually unpleasant clustering. The DTW perfectly clusters the Spanish shields but fails to distinguish the Polish from the French shields. Our shotgun distance in the only one to separate all shields correctly.

4.2 Classification Accuracy

Personalized Medicine: The BIDMC Congestive Heart Failure Database [4] consists of ECG recordings of 15 subjects, which suffer from severe congestive heart failures (Figure 8). The recordings contain noisy or extraneous data, when the recordings started before the machine was connected to the patient. ECG signals show a high level of redundancy due to repetitive heart beats but even a single patient can have multiple different heart beats. To deal with these distortions a classifier has to be invariant to amplitude, uniform scaling, phase shifts and occlusion. The total size of this dataset is equal to 9 million data points (10 hours sampled at at 250 Hz). We used the train/test split provided by [10]. To the best of our knowledge, the best rivalling approach reported a test accuracy of 92.4% [10] and 1-NN DTW scores 62.8%. The shotgun ensemble classifier obtains a much higher test accuracy of 99.3%. This is a result of the design of the shotgun distance: ECG signals are composed of recurring patterns, which are distorted by all kinds of noise. To obtain this score, training the shotgun ensemble classifier took roughly 2 days as all window lengths have to be evaluated. Prediction on the 600 test samples took roughly 1.5 hours in total.

Human Walking Motions: The CMU [5] contains walking motions of 4 subjects. Each motion was categorized by the labels *normal walk* and *abnormal walk* (Figure 8). The data were captured by recording the z-axis accelerometer values of either the right or the left toe. The difficulties in this dataset result from variable-length gait cycles, gait styles and pace due to different subjects throughout different activities including

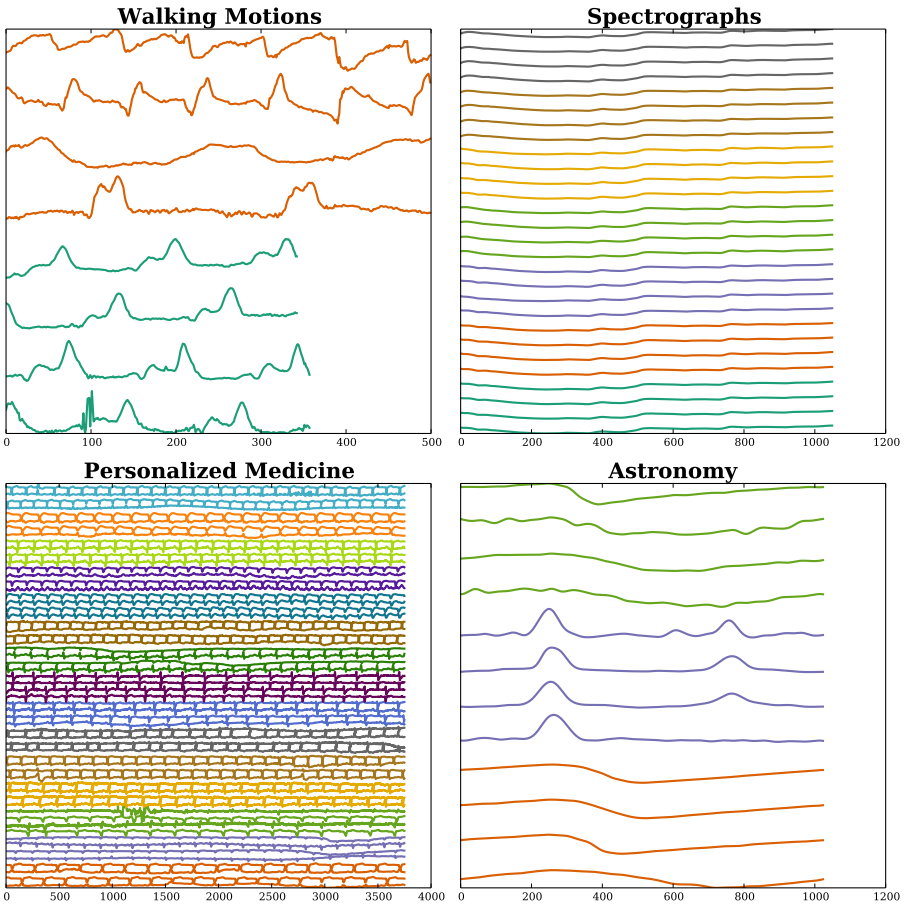


Fig. 8: Four sample representing each class of the case studies: abnormal and normal walking motions, wheat spectrographs, ECG signals, and starlight curves.

stops and turns. To make our results comparable to [21], we used the data provided by their first segmentation approach. We search for normal or abnormal walking patterns. Training the shotgun ensemble classifier took less than a minute. This results in a test classification accuracy of 96.9%, due to the repetitive nature of the data. The accuracy is significantly higher than that of the best rivalling approach in [21] with an accuracy of 91% or 1-NN DTW that scores 66.2%.

Spectrographs: *Wheat* [21] is a dataset of 775 spectrographs of wheat samples grown in Canada. The dataset contains different wheat types like *Canada Western Red Spring, Soft White Spring* or *Canada Western Red Winter* (Figure 8). The class labels define the year in which the wheat was grown. This makes the classification problem much more difficult, as the same wheat types in different years belong to different classes.

The best rivalling approach [21] reported a test accuracy of 72.6% on this dataset and 1-NN DTW obtains a test accuracy of 72.6%. Our shotgun ensemble classifier obtains a significantly higher test accuracy of 80.69%.

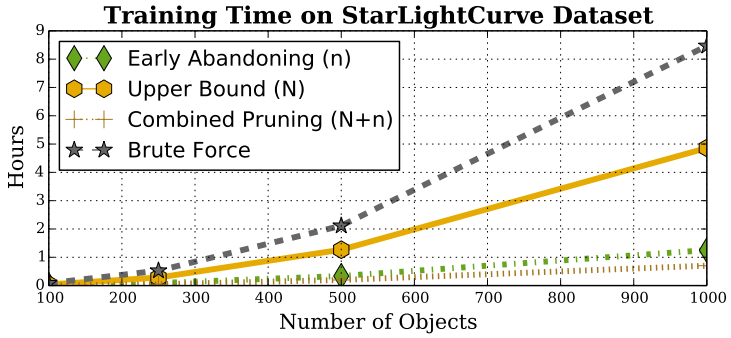


Fig. 9: The time required to execute the shotgun *fit*-method on the *StarLightCurves* dataset using the presented pruning strategies.

Astronomy / Scalability: While it is easy to get large amounts of data, it can be very time consuming to obtain labels for each data item. Thus, it is difficult to obtain large amounts of labelled data. We test the scalability using the largest dataset in the UCR time series archive [3]. This dataset contains three types of star objects: *Eclipsed Binaries*, *Cepheids* and *RR Lyrae Variables*. The *Cepheids* and *RR Lyrae Variables* have a similar shape and are hard to separate (Figure 8 top and bottom). To the best of our knowledge, the highest reported test accuracy is 93.68% [20] with 52 minutes for training and 1-NN DTW scores 90.7%. The test accuracy of our shotgun ensemble classifier is 95.3%.

Scalability: To test the scalability of the shotgun *fit*-method, we iteratively doubled the number of samples from 100 to 1000, each of length 1024, and measured the pruning strategies presented in this paper. Figure 9 shows that the time of the *brute force* algorithm grows quadratically to approximately 9 hours for 1000 samples. *Early abandoning* reduces this by a factor of 7 and in combination with the *upper bound* by a factor of 12 to only 41 minutes. Both pruning strategies combined significantly reduce the run-time for training when the number of samples is increased.

Parameter Space: Figure 10 shows the parameter space for different window lengths and a sample time series for each of the classification case studies. The optimal window lengths correlate with the substructures contained in the time series and no general trend can be observed. The best window length for the astronomy dataset is around 300, for the walking motions around 100, for the spectrographs close to 250, and for the

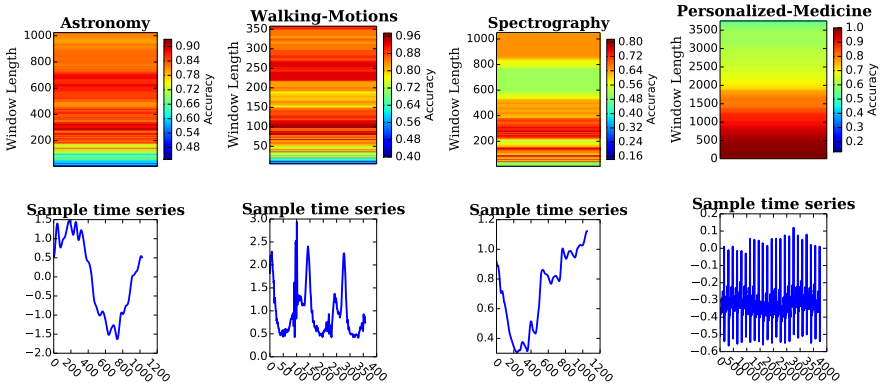


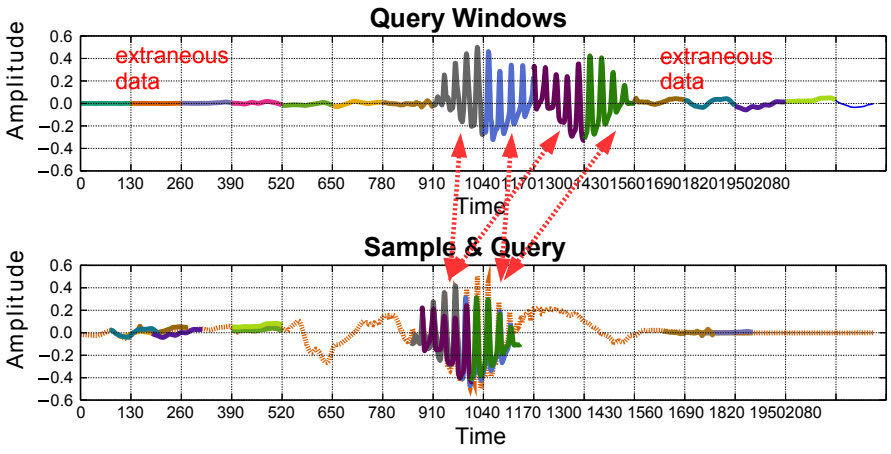
Fig. 10: Top: a heatmap illustrating the optimal window length for each case study. Red indicates the window length with the highest accuracy. Bottom: a sample time series.

personalized medicine around 130. The window length of 100 for the walking motions is roughly equal to the length of one gait cycle. The length of 130 in the personalized medicine dataset correlates with the length of one heartbeat.

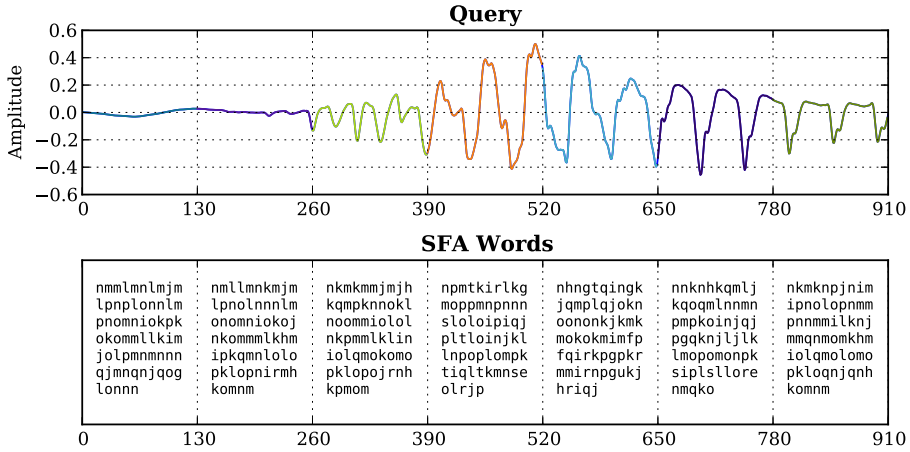
4.3 Computational Bioacoustics:

Producers set up traps in the field that lure and capture pests, in order to detect and count these. Manual inspection of traps is a procedure that is both costly and error prone. Repeated inspection must be carried manually, sometimes in areas that are not easily accessible. A novel recording device has been recently introduced [6]. The core idea is to embed a device in an insect trap to record the fluctuations of light received by a photoreceptor as an insect passes a laser beam and partially occludes light. The samples are recorded at 16 kHz and 1s long but the insect motion within each sample is typically only a few hundredths of a second long. The bandwidth between 0.2-4 kHz is most characteristic. There are three datasets with 5, 9 and 10 insects species [29] available. We focus on the first dataset D1 that was collected from 5 insects, namely: *Aedes aegypti male*, *Fruit flies mixed sex*, *Culex quinquefasciatus female*, *Culex tarsalis female*, *Culex tarsalis male.* It consists of a train/test split with 500 and 5000 recordings. Figure 11a shows two of those insect recordings.

To connect time series analysis with bioacoustics, we use the Symbolic Fourier Approximation (SFA) [24]. Its symbolic and thus compact representation of a time series has shown to be capable of exact similarity search and to index terabyte-sized datasets. Our workflow consists of feature extraction and feature matching. SFA is applied to extract features, which are then passed to the shotgun classifier. SFA reduces noise by the use of low pass filtering and quantization. It can be thought of as the chromatic scale. Pitch levels are mapped to an alphabet of symbols. For example: {C-D-E-F-G-A-H} for the C major scale. In SFA adjacent frequency ranges are mapped to an alphabet of symbols. The SFA transformation results in a character string (see Figure 11b). Each



(a) An insect passes the laser multiple times, causing an *echo*. The shotgun classifier aligns these two signals.



(b) The query is cut into windows, and the SFA representations are calculated.

Fig. 11: Computational Bioacoustics.

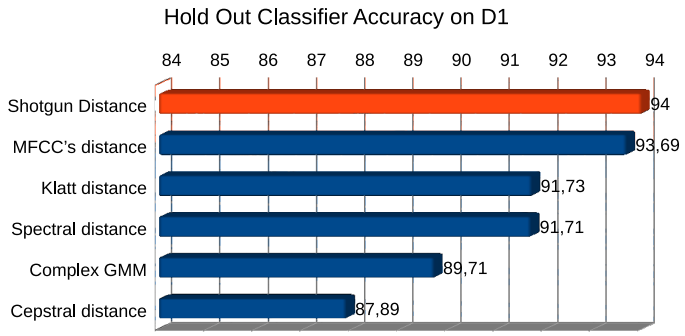


Fig. 12: Classifier accuracies on the test dataset.

symbol represents an interval in the frequency domain. The interested reader is referred to [30, 24] for details on the SFA algorithm.

In particular, noise is generated by the angle and the speed of an insect passing the photoreceptor. This affects the recorded intensities. SFA's noise reduction accounts for these differences in the intensity. By introducing the shotgun classifier for feature matching, we obtain invariance to the time of the insect passage. Shotgun distance further deals with outliers like multiple insects passing the laser within a short time frame (see Figure 11a).

We compared the shotgun distance to common feature vectors in computational bioacoustics like MFCCs, the Klatt spectrum, the Spectral distance, Complex GMM, and Cepstral Distance [29]. Using a small window length of 132, 120 SFA features and 22 SFA symbols performed best. Our approach scores the highest test accuracy with 94% (Figure 12).

This shows that the shotgun distance in combination with SFA are applicable to computational bioacoustics.

4.4 UCR Classification Benchmark Datasets

We used a standardized benchmark [3] to evaluate our approach. There are three types of datasets:

- synthetic: these were created by a scientist.
- real: these were recorded from a sensor.
- shape: representing time series which were generated by processing the shape (contour) of an object.

All samples in these datasets were preprocessed by a domain expert to have an approximate alignment of the objects and fixed length. Each dataset consists of a train and a test subset. The shotgun ensemble classifier is compared to state of the art time series classifiers like shapelets [11], fast shapelets [20], 1-NN classifiers using Euclidean distance or dynamic time warping (DTW) with the optimal warping window, support

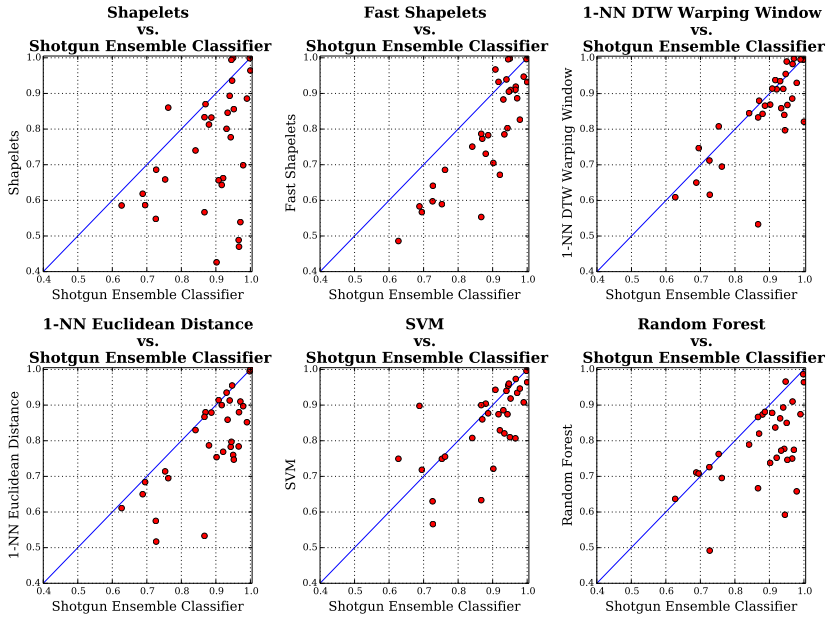


Fig. 13: Accuracy of the shotgun ensemble classifier vs. rivalling approaches.

vector machines (SVM) with a quadratic and cubic kernel, and a tree based ensemble method (random forest). We followed the setup in [11, 20]. The authors in [1] used data transformations (i.e. functional data analysis) to improve classifiers performance. However, they failed to show a significant improvement over 1-NN DTW. We omit data transformations prior to classification for the sake of brevity. The scatter plots in Figure 13 show a pair-wise comparison of each classifier with the shotgun classifier. Each dot represents the test accuracies of the two classifiers on one concrete dataset. Dots below the diagonal line indicate that the shotgun ensemble classifier is more accurate.

The shotgun ensemble classifier is better than fast shapelets and shapelets on the majority of the datasets. We conclude that this is a result of the sensitivity to the overfitting of the shapelet classifiers and the decision tree in particular, where the difference between the train and test accuracy makes up for up to 50 percentage-points. In contrast the shotgun classifier is more robust towards overfitting (see web page [27]). 1-NN DTW is the established benchmark classifier [2]. Our shotgun ensemble classifier is better than 1-NN DTW or 1-NN Euclidean distance on the majority of datasets by a large margin in terms of accuracy. DTW provides invariance to local scaling (time warping). Shotgun distance does not explicitly provide this invariance. However, the results imply that either (a) most UCR datasets do not require local scaling or (b) the shotgun distance provides some local scaling invariance. This will be part of future work. The shotgun distance is equal to Euclidean distance, if the window length is equal to the query length. Thus, the shotgun ensemble classifier performs better than the 1-NN Euclidean distance. SVMs and the shotgun ensemble classifier complement each other

quite well as one classifier is good on a dataset in which the other performs badly. So, at least for datasets which were preprocessed for approximate alignment and fixed length, the choice of the classifier depends on the dataset. When comparing the shotgun ensemble classifier with random forests, the results suggest that the former is more accurate by a large margin. Note that the UCR datasets were preprocessed for approximate alignment and length. Still our shotgun classifier performs significantly better than rivalling state of the art classifiers on a majority of datasets.

4.5 Impact of Design Decisions

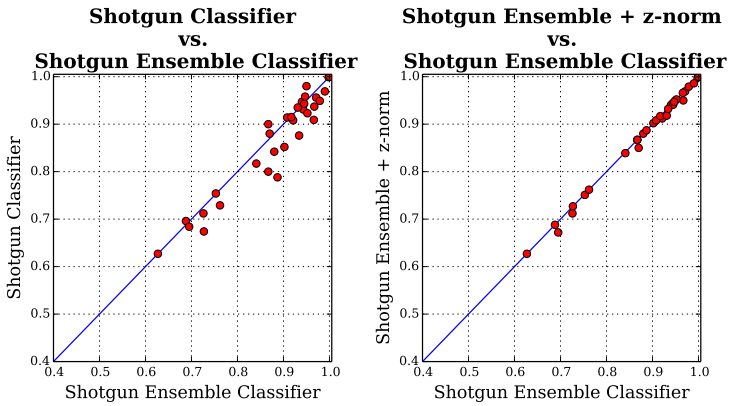


Fig. 14: Classifier accuracies on test subsets for the shotgun ensemble classifier.

The shotgun classifier is based on two design decisions:

1. Building an *ensemble* of shotgun classifiers.
2. *Mean normalization* as a parameter as opposed to always normalizing the mean of all windows.

We chose to use 1-NN classification as it doesn't introduce any new parameters for model training which allows us to focus on the shotgun ensemble classifier. Overall the shotgun ensemble classifier showed a better or equal accuracy on 22 out of 32 datasets when compared to the shotgun classifier using one fixed window length (Figure 14). It improves by up to 10 percentage points on the Motes dataset. As for mean normalization the accuracies increased by up to 2.3 percentage points when treated as a parameter.

5 Conclusion

The time series classification task is complicated by noise, dropouts, subtle distinctions, variable lengths or extraneous data. The shotgun distance is a novel distance measure

based on the characteristic patterns in time series. Shotgun distance utilizes time segments which are vertically and horizontally aligned and scaled between the query and a sample, and thereby simplifying the preprocessing. Based on an ensemble of 1-nearest-neighbor classifiers the shotgun ensemble classifier is presented. To deal with the increased complexity, two pruning strategies for the length and the number of time series are presented. This reduces the computational complexity by one order of magnitude. In our experimental evaluation we show that the shotgun distance performs better than rivalling methods in the context of hierarchical clustering and classification for use cases in computational bioacoustics, human motion detection, spectrographs, astronomy, or personalized medicine. This is underlined by the best classification accuracy on standardized benchmark datasets.

Acknowledgements This project was motived and partially funded by the German Federal Ministry of Education and Research through the project "Berlin Big Data Center (BBDC)". The author would like to thank C. Eichert-Schäfer, F. Schintke, F. Wende, and S. Dreßler for their comments and the dataset owners.

References

1. A. Bagnall, L. M. Davis, J. Hills, and J. Lines, "Transformation Based Ensembles for Time Series Classification," in *SDM*, vol. 12. SIAM, 2012, pp. 307–318.
2. H. Ding, G. Trajcevski, P. Scheuermann, X. Wang, and E. Keogh, "Querying and mining of time series data: experimental comparison of representations and distance measures," *Proceedings of the VLDB Endowment*, vol. 1, no. 2, pp. 1542–1552, 2008.
3. E. Keogh, X. Xi, L. Wei, and C. A. Ratanamahatana, "UCR Time Series Classification/Clustering Homepage." [Online]. Available: http://www.cs.ucr.edu/~eamonn/time_series_data
4. BIDMC. [Online]. Available: http://www.physionet.org/physiobank/database/chfdb/
5. CMU Graphics Lab Motion Capture Database. [Online]. Available: http://mocap.cs.cmu.edu/
6. UCR Insect Contest, 2012. [Online]. Available: http://www.cs.ucr.edu/~eamonn/CE
7. G. Batista, X. Wang, and E. J. Keogh, "A Complexity-Invariant Distance Measure for Time Series," in *SDM*, vol. 11. SIAM / Omnipress, 2011, pp. 699–710.
8. S. Zhang, C. Zhang, and Q. Yang, "Data Preparation for Data Mining." *Applied Artificial Intelligence*, vol. 17, no. 5-6, pp. 375–381, 2003.
9. P. Perner, *Data mining on multimedia data*. Springer, 2002, vol. 2558.
10. B. Hu, Y. Chen, and E. Keogh, "Time Series Classification under More Realistic Assumptions," in *SDM*. SIAM, 2013, pp. 578–586.
11. A. Mueen, E. J. Keogh, and N. Young, "Logical-shapelets: an expressive primitive for time series classification," in *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '11. ACM, 2011, pp. 1154–1162.
12. Kaggle: Go from Big Data to Big Analytics. [Online]. Available: https://www.kaggle.com
13. P. Schäfer, "Towards time series classification without human preprocessing," in *MLDM 2014*, ser. Lecture Notes in Computer Science, P. Perner, Ed., vol. 8556, 2014, pp. 228 – 242.
14. J. C. Venter and Others, "The Sequence of the Human Genome," *Science*, vol. 291, no. 5507, pp. 1304–1351, 2001.
15. E. Keogh and S. Kasetty, "On the need for time series data mining benchmarks: a survey and empirical demonstration," in *KDD*. ACM, 2002, pp. 102–111.

16. J. Lin, R. Khade, and Y. Li, "Rotation-invariant similarity in time series using bag-of-patterns representation." *J. Intell. Inf. Syst.*, vol. 39, no. 2, pp. 287–315, 2012.
17. T. Warren Liao, "Clustering of time series data—a survey," *Pattern Recognition*, vol. 38, no. 11, pp. 1857–1874, 2005.
18. T. Rakthanmanon, B. Campana, A. Mueen, G. Batista, B. Westover, Q. Zhu, J. Zakaria, and E. Keogh, "Searching and mining trillions of time series subsequences under dynamic time warping," Y. Qiang, D. Agarwal, and J. Pei, Eds., ACM. ACM, 2012, pp. 262–270.
19. H. Sakoe and S. Chiba, "Dynamic programming algorithm optimization for spoken word recognition," *IEEE Trans. Acoust., Speech, Signal Processing*, no. 1, pp. 43–49, 1978.
20. T. Rakthanmanon and E. Keogh, "Fast Shapelets: A Scalable Algorithm for Discovering Time Series Shapelets," in *SDM*. SIAM, 2013, pp. 668–676.
21. L. Ye and E. J. Keogh, "Time series shapelets: a novel technique that allows accurate, interpretable and fast classification," *DMKD*, vol. 22, no. 1-2, pp. 149–182, 2011.
22. R. Agrawal, C. Faloutsos, and A. Swami, "Efficient similarity search in sequence databases," *Foundations of Data Organization and Algorithms*, vol. 730, pp. 69–84, 1993.
23. Q. Chen, L. Chen, X. Lian, Y. Liu, and J. X. Yu, "Indexable PLA for Efficient Similarity Search." in *VLDB*. ACM, 2007, pp. 435–446.
24. P. Schäfer and M. Högqvist, "SFA: a symbolic fourier approximation and index for similarity search in high dimensional datasets," in *EDBT*, E. A. Rundensteiner, V. Markl, I. Manolescu, S. Amer-Yahia, F. Naumann, and I. Ari, Eds. ACM, 2012, pp. 516–527.
25. J. Lin, E. J. Keogh, L. Wei, and S. Lonardi, "Experiencing SAX: a novel symbolic representation of time series," *Data Mining and Knowledge Discovery*, vol. 15, no. 2, 2007.
26. C. Lipowsky, E. Dranischnikow, H. Göttler, and T. Gottron, "Alignment of Noisy and Uniformly Scaled Time Series," in *DEXA*. Springer, 2009, pp. 675–688.
27. Shotgun Distance Webpage. [Online]. Available: http://www.zib.de/patrick.schaefer/shotgun
28. L. Ye and E. J. Keogh, "Time series shapelets: a new primitive for data mining," in *Proceedings of the 15th ACM SIGKDD*. ACM, 2009, pp. 947–956.
29. I. Potamitis and P. Schäfer, "On classifying insects from their wing-beat: New results," *Ecology and acoustics: emergent properties from community to landscape, Paris, France*, 2014.
30. P. Schäfer and S. Dreßler, "Shooting Audio Recordings of Insects with SFA," in *AmiBio Workshop, Bonn, Germany*, 2013.