Towards Time Series Classification without Human Preprocessing

Patrick Schäfer

Zuse Institute Berlin
patrick.schaefer@zib.de

Abstract Over the past decade time series classification has been attracting increasing attention by the research community. Traditional algorithms were designed to work on clean data, which have been preprocessed by a human domain expert to remove duplicates and extract interesting patterns of equal length and scaling. There is a need for algorithms that are able to process raw input data 'as-is', as human preprocessing is not feasible in terms of monetary resources. Shotgun distance mimics the task of human preprocessing by extracting and aligning z-normalized time segments from a query to a sample. A time series is represented by multiple time segment lengths as part of our shotgun ensemble classifier. Our classifier improves the accuracy on case studies in the context of bioacoustics, human motion detection, spectrographs or personalized medicine. It further performs better than state of the art on the UCR classification benchmark datasets.

1 Introduction

Time series result from recording data over time. The time series classification task aims at assigning a class label to a time series. For this the distinguishing features (the model) between the class labels are trained on a train dataset. When an unlabeled query time series is recorded, the model is used to determine to which class the time series belongs. The classification of time series has gained increasing interest over the past decade [1,2,3]. Most research on time series classification algorithms assumes that the data is segmented so that interesting patterns are aligned and data frames have the same length and scaling. Sensors and other input sources, in contrast, may produce data of variable length, noisy data with dropouts and extraneous sections, which are highly redundant due to the recurring patterns. These signals include ECG [4] or EEG signals from personalized medicine, human walking motions [5], wing beats from flying insects [6] or the symmetry in the shape of objects.

Empirical evaluation suggests that classifiers based on 1-nearest-neighbour Euclidean distance or dynamic time warping (DTW) are hard to beat [1,2]. However, these methods align two entire time series to calculate their similarity. Therefor the data has to be preprocessed by a domain expert by hand to filter the data and extract equal-length, equal-scale, and aligned patterns. This significantly eases the subsequent data mining task both in terms of the cost of the execution time and the complexity of the algorithm. However, this preprocessing task is too time consuming in terms of human resources. Only few algorithms exist that deal with unprocessed time series data. These are based on matching time series by their structural similarity [7,8,9].

As traditional data mining algorithms were not designed for real world datasets, a multitude of international competitions were hosted [10,6,11]. These included the task of identifying and classifying whale calls [10], human walking motions [10] and flying insects [6], to name but a few.

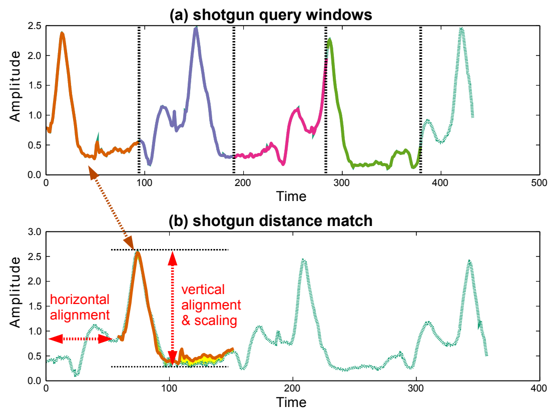


Figure 1: The shotgun distance consists of horizontal and vertical alignment and scaling of each query subsequences.

This work introduces a novel similarity metric for mining unaligned, unscaled, raw time series datasets. *Shotgun distance* (Figure 1) vertically and horizontally aligns time segments (subsequences) of a query to a sample time series, and thereby avoids preprocessing the data by a human. This is achieved by breaking the query into disjoint subsequences of fixed length first. Next, each query subsequence is slid along the sample to find the best matching position in terms of minimizing a distance metric (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. Z-normalization is applied prior to each distance computation, to provide the same vertical alignment and scaling of each subsequence. The *shotgun ensemble classifier* is based on an ensemble of 1-NN classifiers using the shotgun distance at multiple time segment lengths. Our contributions are as follows:

– We present the shotgun distance motivated by the challenges arising from the classification of datasets, which were not preprocessed by a human, in Section 2.
– We introduce the shotgun distance that mimics human preprocessing in Section 3.
– We present the shotgun ensemble classifier, which represents a time series at multiple subsequences lengths in Section 3.4.
– Two pruning strategies are presented which reduce the computational complexity of the shotgun classifier by one order of magnitude in Section 3.5.
– We show that the shotgun ensemble classifier is significantly more accurate than rivalling state of the art approaches based on 5 case studies and the standardized benchmark datasets in Section 4.

2 Background & Motivation

The utility of shotgun distance is based on the assumption that a signal is composed of characteristic patterns. There is a multitude of signals, which are composed of patters like ECG and EEG recordings, shape-based signals, recordings in bioacoustics or accelerometer data (compare Section 4.1).

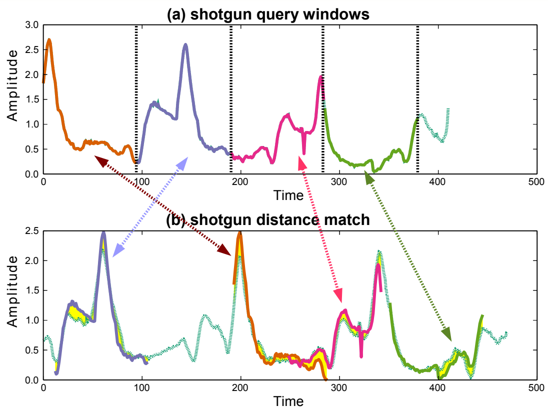


Figure 2: Matching the gait cycles in the query to the sample is complicated due to different amplitudes, phase-shifts, variable lengths and noise.

As a concrete example, consider a human motion from the CMU Graphics Lab Motion Capture Database (CMU) [5]. Each motion was categorized by the labels *normal walk* and *abnormal walk*. 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 including stops and turns. Figure 2 illustrates the walking motion of a subject, which contains 4 gait cycles. The classification of a query is difficult, as the samples are not preprocessed to have an approximate alignment, length, scale or number of gait cycles.

Shotgun distance mimics human preprocessing by vertically aligning and horizontally scaling the query to a sample (Figure 1). First, the query is broken into disjoint subsequences of fixed length, hereafter referred to as windows. Next, each query window is slid along the sample to find the best matching position by minimizing the Euclidean distance (horizontal alignment). To scale the query window to the sample, z-normalization is applied prior to the Euclidean distance computations. Figure 2 (bottom) illustrates the result of this process. The gait cycles perfectly match the sample, even though these differ in scale, have a variable length, and a phase-shift and noise occurs.

The quality of the shotgun distance is subject to two parameters (Figure 1):

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.

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 and matched to the sample. This mostly happens with signals which were preprocessed by a human for alignment and length. In contrast, human motions contain repetitive gait cycles. Matching 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.

3 Shotgun Distance

3.1 Definitions

A time series consists of a sequence of real values:

$$T = (t_1, \ldots, t_n)$$

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 $S_{i;w} = (t_i, \ldots, t_{i+w-1})$ of length w using a windowing function. Two consecutive windows can* overlap *within an interval of* $[0, w)$*:*

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

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

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

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 metric. The shotgun distance is a distance metric that minimizes the Euclidean distance between each disjoint window in the query Q and the sliding windows in a sample S. Intuitively, each gait cycle is slid along a longer walking motion to find the best matching positions (compare Figure 1).

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 $Q_a \epsilon \hat{\omega}(Q, w, 0) = Qs$ and the sliding sample windows $S_b \epsilon \hat{\omega}(S, w, w-1) = Ss$:*

$$D_{shotgun}(Q, S) = \sum_{a=1}^{len(Qs)} \min\{D(Q_a, S_b) \mid S_b \epsilon Ss\}$$

Algorithm 1 The shotgun distance.

```
double ShotgunDistance(query,sample,W_SIZE,M_NORM)
(1) totalDist = 0.0
    // search for each disjoint query window
(2) for q in disjoint_windows(query,W_SIZE,M_NORM)
(3)   qDist = MAX_VALUE
      // search for the best matching sliding window
(4)   for s in sliding_windows(sample,W_SIZE,M_NORM)
(5)     qDist = min(qDist,EuclideanDist(q,s))
(6)   totalDist += qDist
(7) return totalDist
```

This definition of similarity resembles the task of preprocessing of raw time series data, which is otherwise carried out by a human. Characteristic patterns (i.e. the gait cycles) are extracted, scaled and the cycles are aligned. The latter provides invariance to the time ordering of the patterns and allows for comparing variable length time series. For n equal to w, the shotgun distance is equal to the Euclidean distance.

The shotgun distance metric allows for searching for the most similar sample time series (the 1-nearest-neighbor) to a query time series and labeling the query by this sample.

3.2 Shotgun Distance Algorithm

The brute-force shotgun distance algorithm is described in Algorithm 1. It makes use of the Euclidean distance, and can be tuned by the two parameters *window length* W_SIZE and *mean* M_NORM (the standard deviation of q and s is always normed to 1 regardless of M_NORM). It first obtains disjoint query windows (line 2) and searches for the best sliding window, i.e., the position in the sample that minimizes the Euclidean distance (line 4-5). Finally, the distances are accumulated for each query window (line 6).

Time 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 a constant window length w:

$$\begin{aligned} O(\text{Shotgun}) &= O\left(\underbrace{\frac{|Q|}{w}}_{\text{disjoint windows}} \cdot w \cdot \underbrace{(|S| - w + 1)}_{\text{sliding windows}}\right) \\ &\quad \text{for } n = max(|Q|, |S|) \\ &\Rightarrow O\left(n^2 - nw\right) \end{aligned}$$

Note, that for large window lengths $w \sim n$ this complexity is close to linear in n. For small window lengths $w \ll n$ the complexity is quadratic in n^2.

Algorithm 2 The Shotgun Ensemble Classifier.

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

String predictEnsemble(query,samples,bestScore,windows,M_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,M_NORM)
(5) return most frequent label from windowLabels

[(int,int)] fit(samples,labels,M_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,M_NORM)
(6)     if (nnLabel==query.label) correct++
      // store scores for each window length
(7)   scores.push((correct, len))
(8) return scores
```

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 a set of samples using the shotgun distance (line 3-5). Finally, the query is labeled by the class label of the 1-NN *nn*.

The *fit*-method (Algorithm 2) is applied to maximize the accuracy of the train samples by the use of leave-one-out cross-validation (lines 4-8). It records the accuracies for all window lengths starting from the *maxLen* (the length of the longest time series) down to *minLen* (lines 2-7). The M_NORM-parameter is a Boolean parameter, which is constant for a whole dataset and not set per sample. It depends on the characteristics of the dataset.

3.4 Shotgun Ensemble Classifier

By intuition every dataset consists of substructures at multiple window lengths caused by different walking motions, heart beats, duration of vocals, length of shapes, to name but a few examples. For example, each human may have a different length of a gait cycle. Thus, we represent each sample by a set of window lengths.

Using the *predictEnsemble*-method (Algorithm 2), a label is determined for the best window lengths. Using a constant parameter $factor\epsilon[0, 1]$ and the best

Algorithm 3 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
(3)   if (dist > bestDist) return MAX_VALUE           // early abandoning
(4) return dist

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

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

accuracy *bestScore* obtained from the train samples, the best window lengths are given by: $correct > bestScore \cdot factor$ (line 3). Finally, the most frequent class label is chosen from the set of labels.

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 set 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 $[0.95, 1]$ 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 [7,12,13].

Early Abandoning: The purpose of the *ShotgunDistance*-method (Algorithm 3) 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 Use an upper bound on the current accuracy.

```
[(int,int)] fit(samples,labels,M_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,M_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: correct $\leq$ (current correct $+$ remaining samples) $= N$.

Thus, we do not need to obtain the exact accuracy for a window length in Algorithm 4 (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 ensemble classifier is underlined by case studies and the UCR time series classification benchmark datasets [3]. Each dataset is split into two subsets: *train* and *test*. By the use of the same train/test splits the results are comparable to the previously published ones [1,2,7,13,14,15]. **The train subset is used for parameter and model selection**. **The test subset is used to test the accuracy of the classifiers solely.** Our web page [16] contains a spreadsheet with all raw numbers, the shotgun ensemble classifier source code, and the python code to train and test the more complex SVM and random forests classifiers. All benchmarks were performed on a shared memory machine running Linux with 8 Quad-Core AMD Opteron 8358 SE processors and Java JDK x64 1.7.

4.1 Case Studies

Astronomy / Scalability: We test the scalability using the largest dataset in the UCR time series archive [3]. It contains three types of star objects: Eclipsed Binaries, Cepheids and RR Lyrae Variables. To test the scalability of the shotgun *fit*-method, we used the *train* subset containing 1000 time series each of length 1024. The size of the dataset was iteratively doubled, starting from 100 to 1000 samples. We measured the different pruning strategies presented in this paper on the length n and the number of samples N.

Dataset	**Rivalling Method**	**Shotgun Ensemble Classifier**	
Personalized Medicine	92.4% [8]	99.3%	$factor = 1$, mean=true
Walking Motions	91% [17]	96.9%	$factor = 0.95$, mean=true
Spectrographs	72.6% [12]	80.7%	$factor = 0.95$, mean=true
Bio Acoustics	93.29% [6]	92.38%	$factor = 1$, mean=true
Astronomy	93.68% [13]	95.3%	$factor = 0.97$, mean=true

Table 1: Test accuracies on the case studies.

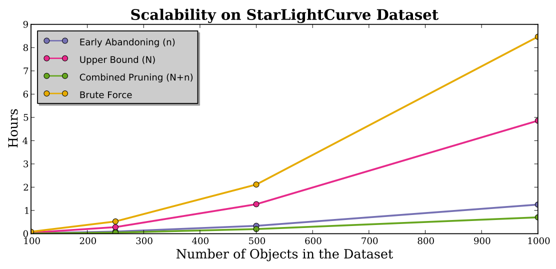


Figure 3: The time required to execute the shotgun *fit*-method on the *StarLightCurves* dataset using the presented pruning strategies.

To the best of our knowledge, the highest reported test accuracy is 93.68% [13] with a run-time of 52 minutes for training. The test accuracy of the shotgun ensemble classifier is 95.3% when using *factor* = 0.97 and *mean* = *true*. The results in Figure 3 show that the elapsed time of the *brute force* algorithm grows quadratically with the number of objects, requiring approximately 9 hours for 1000 objects. *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.

Personalized Medicine: The BIDMC Congestive Heart Failure Database [4] consists of ECG recordings of 15 subjects, which suffer from severe congestive heart failures. 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 in [8], which selected 150 minutes for training and 450 minutes for testing and search for individual patient heart beats (15 distinct classes). There are 600 samples for training at length 3750 and 600 samples for testing at length 11250.

To the best of our knowledge, the best rivalling approach reported a test accuracy of 92.4% on this dataset [8]. The shotgun ensemble classifier obtains a much higher test accuracy of 99.3%, when using *factor* = 1 and *mean* = *true*.

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. Testing all 600 samples took roughly 1.5 hours.

Human Walking Motions: The CMU [5] contains walking motions of 4 subjects. Each motion was categorized by the labels *normal walk* and *abnormal walk*. 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 stops and turns. To deal with these distortions, a classifier needs to be invariant to amplitudes, uniform scaling, phase shifts and occlusions. To make our results comparable to [17], we used the data provided by their first segmentation approach. The dataset contains 40 samples for training and 228 samples for testing, each of variable lengths with an upper limit of 500 points. We search for normal or abnormal walking patterns, representing the two distinct classes.

Training the shotgun ensemble classifier took less than a minute. This results in a test classification accuracy for the shotgun classifier of 96.9% when using $factor = 0.95$ and $mean = false$. The accuracy is significantly higher than that of the best rivalling approach in [17] which reports an accuracy of 91%.

Spectrographs: *Wheat* [12] is dataset of 775 spectrographs of wheat samples grown in Canada. The data is split into 49 samples of length 1050 for training and 726 samples of length 1050 for testing. The dataset contains different wheat types like Canada Western Red Spring, Soft White Spring or Canada Western Red Winter. The class label define the year in which the wheat was grown. This makes the classification problem difficult, as the same wheat types in different years belong to different classes.

The best rivalling approach [12] reported a test accuracy of 72.6% on this dataset and the 1-NN Euclidean distance classifier obtains a test accuracy of 66.9%. Our shotgun ensemble classifier obtains a much higher test accuracy of 80.69%, when using $factor = 0.95$ and $mean = true$.

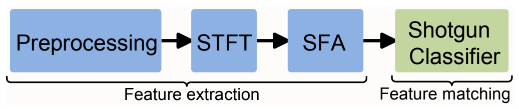


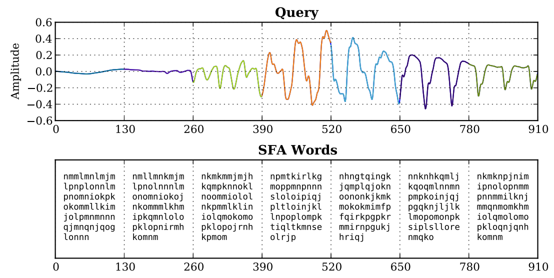
Figure 4: Insect Classification Workflow.

Bioacoustics: The classification of audio recordings to measure the occurrences of a species has been employed as one indicator of biodiversity. The University of California Riverside (UCR) built sensors to collect data from flying insects by an optical sensor [6]. The sensor detects the speed and wing beats of flying insects. The insect signal, characterized by a distinctive buzz, is typically only a few hundredths of a second long. The samples for the insect motions were recorded at 16 kHz and are 1s long. The bandwidth between 0.2-4 kHz is most characteristic.

To connect time series analysis with bioacoustics, we use Symbolic Fourier Approximation (SFA) [18]. 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. The workflow consists of feature extraction and feature matching and is represented in Figure 4. SFA is applied to extract features, which are then passed to the shotgun classifier.

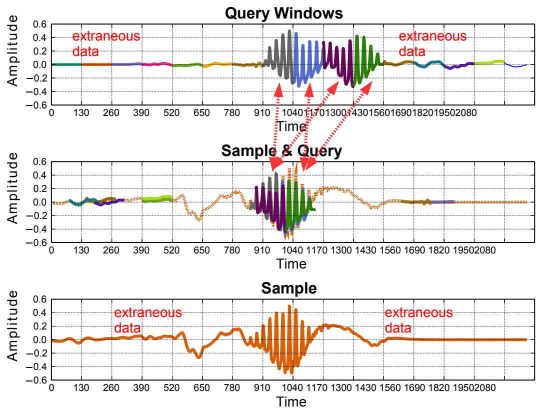
The solution utilizes SFA to reduce noise by the use of quantization applied on top of the Fourier-transformation. SFA can be thought of as the chromatic scale. Pitch levels are mapped to a finite alphabet of symbols. For example: {C-D-E-F-G-A-H} for the C major scale. The SFA transformation results in a character string. Each symbol represents an interval in the frequency domain (see Figure 5a).

In particular, noise is generated by the angle or the speed of an insect passing a sensor. This affects the intensity of the recorded signal. SFA's noise reduction accounts for these differences in the recorded intensities. By introducing the shotgun classifier for feature matching, the horizontal alignment of the samples is simplified. This is mainly caused by the invariance to the time ordering when matching two signals. Shotgun distance further deals with outliers like multiple insects passing the laser within a short time frame (see



```
nmmlmnlmjm   nmllmnkmjm   nkmkmmjmjh   npmtkirlkg   nhngtqingk   nnknhkqmlj   nkmknpjnim
lpnplonnlm   lpnolnnnlm   kqmpknnokl   moppmnpnnn   jqmplqjokn   kqoqmlnnmn   ipnolopnmm
pnomniokpk   onomniokoj   noommiolol   sloloipiqj   oononkjkmk   pmpkoinjqj   pnnmmilknj
okommllkim   nkommmlkhm   nkpmmlklin   pltloinjkl   mokokmimfp   pgqknjljlk   mmqnmomkhm
jolpmnmnnn   ipkqmnlolo   iolqmokomo   lnpoplompk   fqirkpgpkr   lmopomonpk   iolqmolomo
qjmnqnjqog   pklopnirmh   pklopojrnh   tiqltkmnse   mmirnpgukj   siplsllore   pkloqnjqnh
lonnn        komnm        kpmom        olrjp        hriqj        nmqko        komnm
```

(a) The query (top) is cut into windows, and a SFA representation (bottom) is calculated using an alphabet of size 26.



(b) An insect passes the laser multiple times, causing an *echo*. The shotgun classifier aligns these two signals by matching the distinctive sound.

Figure 5

Figure 5b). For the sake of brevity, the interested reader is referred to [19] for details on the approach and SFA.

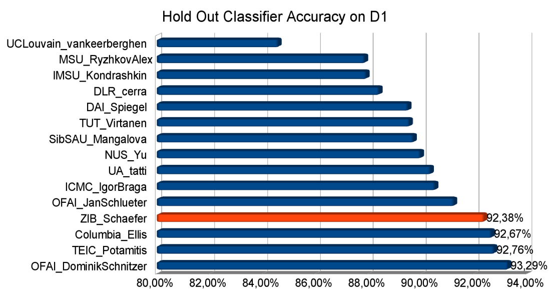


Figure 6: Classifier accuracies on the D1 test dataset. ZIB_Schaefer represents the shotgun classifier.

We focus on the dataset D1, which was collected from 5 distinctive classes and consists of 5000 objects, whereas a subset of 500 samples were used to train the classifier. Using a small window length of 130 and a small number of SFA symbols of 8-10 performed best. Our approach scored within 1 percentage point of the best approach (Figure 6).

These results show that time series analysis is applicable to computational bioacoustics and that our approach competes with the rivalling methods applied in the UCR insect contest.

4.2 UCR classification benchmark datasets

We evaluated our shotgun ensemble classifier using a standardized benchmark [3]. Each dataset consists of a train and a test subset. The shotgun ensemble classifier with $factor = 0.95$ and $mean = [true, false]$ (see [16]) is compared to state-of-the-art classifiers in the context of time series classification like shapelets [7], fast shapelets [13] and 1-NN classifiers using Euclidean distance or dynamic time warping (DTW) with a warping window. Complex classifiers such as support vector machines (SVM) with a quadratic and cubic kernel and tree based ensemble methods such as random forests were benchmarked, too. We follow the setup of the experiments in [13] and [7]. Our web page [16] contains all raw numbers.

To give an intuitive illustration of the performance of the classifiers, scatter plots for a pair-wise comparison of the classifiers with the shotgun ensemble classifier are presented. In the scatter plot the test accuracies of the two classifiers are represented by one dot, each dot representing one concrete dataset. The further a dot is located from the line, the greater the difference in accuracy of the two classifiers. If there are more dots on one side of the diagonal, then one classifier is more precise than the other in a majority of datasets.

Figure 7 shows that the shotgun ensemble classifier is better than fast shapelets and shapelets on the majority of the datasets presented in this paper. We conclude that this is a result of the overfitting of the shapelet classifiers. The difference between the train and test accuracy makes up for up to 50 percentage-points for shapelet-based classifiers. In contrast the shotgun classifier is more robust towards overfitting (see raw data [16]).

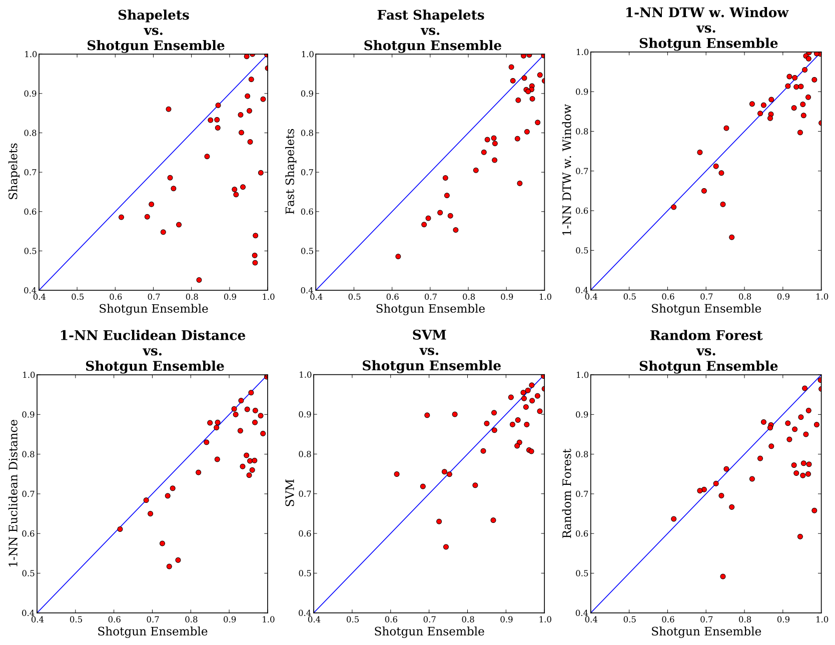


Figure 7: Accuracy of the shotgun ensemble classifier vs. rivalling approaches.

1-NN DTW is known to be a very competitive classifier and has been widely used for time series classification [2]. Shotgun ensemble classification is better than 1-NN DTW or 1-NN Euclidean distance on the majority of datasets by a large margin in terms of accuracy. This implies that either (a) the datasets do not require local scaling (warping) or (b) the shotgun ensemble classifier provides this kind of invariance. This will be part of future work. Recall that 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 seem to complement each other quite well in the sense that 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.

We conclude that the shotgun ensemble classifier is a very competitive classifier for time series classification. In a majority of datasets the shotgun ensemble classifier is significantly more accurate than the rivalling methods.

5 Related Work

There is a multitude of literature on time series classification and mining including [1,2,3]. This includes benchmarks for SVMs and random forests, too, as well as classification based on matching the shape of the entire time series using a distance measure like the Euclidean distance or dynamic time warping. The closest work to the shotgun ensemble classifier is that of shapelets [7] and fast shapelets [13], which use subsequences for classification, too. In contrast to the shotgun classifier, these approaches create a decision tree based on representative variable-length subsequences (shapelets) and use thresholds for branching. Today, only few algorithms exist that deal with time series classification without human preprocessing. One notable exception is [8], but even here it remains unclear how to cope with the increased time complexity compared to traditional data mining algorithms. Moreover, the authors did not present a general model for training the parameters of their classifier. The shotgun classifier is inspired by shotgun sequencing. The latter is a technique introduced in bioinformatics to find an alignment of two DNA or protein sequences [20]. These sequences are represented as a character string over a finite alphabet of characters, rather than numerical features. Shotgun sequencing has been applied to find the horizontal displacements in the production of steel coils [21]. To find the horizontal displacement the authors use the median on the differences of the calculated starting positions for every pair of subsequences.

6 Conclusion

The time series classification task is complicated by noise, dropouts, subtle distinctions, variable lengths or extraneous data. Time consuming human preprocessing is needed to filter the 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 mimics human 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. The experimental evaluation shows that the shotgun ensemble classifier performs better than rivalling methods in the context of computational bioacoustics, human motion detection, spectrographs, astronomy, or personalized medicine. This is underlined by the best classification accuracy on the UCR time series classification datasets when compared to state of the art.

Acknowledgement

The author would like to thank Florian Schintke, Florian Wende and Sebastian Dreßler for their valuable comments on the paper and the owners of the datasets for making their data available.

References

1. A. Bagnall, L. M. Davis, J. Hills, and J. Lines, “Transformation Based Ensembles for Time Series Classification,” in *SDM*, 2012.
2. H. Ding, G. Trajcevski, P. Scheuermann, X. Wang, and E. J. Keogh, “Querying and mining of time series data: experimental comparison of representations and distance measures,” *Journal Proceedings of the VLDB Endowment*, 2008.
3. E. Keogh, Q. Zhu, B. Hu, Y. Hao, X. Xi, L. Wei, and C. A. Ratanamahatana, “UCR Time Series Classification/Clustering Homepage.” `http://www.cs.ucr.edu/~eamonn/time\_series\_data/`.
4. The BIDMC congestive heart failure database. `http://www.physionet.org/physiobank/database/chfdb/`.
5. CMU Graphics Lab Motion Capture Database. `http://mocap.cs.cmu.edu/`.
6. UCR Insect Contest. `http://www.cs.ucr.edu/~eamonn/CE/`, 2012.
7. A. Mueen, E. J. Keogh, and N. Young, “Logical-shapelets: an expressive primitive for time series classification,” in *KDD*, ACM, 2011.
8. B. Hu, Y. Chen, and E. Keogh, “Time Series Classification under More Realistic Assumptions,” in *SDM*, 2013.
9. J. Zakaria, A. Mueen, and E. J. Keogh, “Clustering Time Series Using Unsupervised-Shapelets,” in *ICDM*, IEEE Computer Society, 2012.
10. Kaggle: Go from Big Data to Big Analytics. `https://www.kaggle.com/`.
11. MQUTeR Environmental Workbench. `http://sensor.mquter.qut.edu.au/`.
12. L. Ye and E. J. Keogh, “Time series shapelets: a new primitive for data mining,” in *KDD*, ACM, 2009.
13. T. Rakthanmanon and E. Keogh, “Fast Shapelets: A Scalable Algorithm for Discovering Time Series Shapelets,” in *SDM*, 2013.
14. G. Batista, X. Wang, and E. J. Keogh, “A Complexity-Invariant Distance Measure for Time Series,” in *SDM*, 2011.
15. Fast Shapelet Results. `http://alumni.cs.ucr.edu/~rakthant/FastShapelet/`.
16. The Shotgun Distance Webpage. `http://www.zib.de/patrick.schaefer/shotgun/`.
17. L. Ye and E. J. Keogh, “Time series shapelets: a novel technique that allows accurate, interpretable and fast classification,” *Data Min. Knowl. Discov.*, 2011.
18. P. Schäfer and M. Högqvist, “SFA: a symbolic fourier approximation and index for similarity search in high dimensional datasets,” in *EDBT*, ACM, 2012.
19. P. Schäfer and S. Dreßler, “Shooting Audio Recordings of Insects with SFA,” in *to appear, AmiBio Workshop, Bonn, Germany*, 2013.
20. J. C. Venter and Others, “The Sequence of the Human Genome,” *Science*, 2001.
21. C. Lipowsky, E. Dranischnikow, H. Göttler, T. Gottron, M. Kemeter, and E. Schömer, “Alignment of Noisy and Uniformly Scaled Time Series,” in *DEXA*, Lecture Notes in Computer Science, Springer, 2009.