Learning Time-Series Shapelets

Josif Grabocka, Nicolas Schilling, Martin Wistuba, Lars Schmidt-Thieme
Information Systems and Machine Learning Lab
University of Hildesheim
Samelsonplatz 22, Hildesheim 31141, Germany
{ josif, schilling, wistuba, schmidt-thieme }@ismll.uni-hildesheim.de

ABSTRACT

Shapelets are discriminative sub-sequences of time series that best predict the target variable. For this reason, shapelet discovery has recently attracted considerable interest within the time-series research community. Currently shapelets are found by evaluating the prediction qualities of numerous candidates extracted from the series segments. In contrast to the state-of-the-art, this paper proposes a novel perspective in terms of learning shapelets. A new mathematical formalization of the task via a classification objective function is proposed and a tailored stochastic gradient learning algorithm is applied. The proposed method enables learning near-to-optimal shapelets directly without the need to try out lots of candidates. Furthermore, our method can learn true top-K shapelets by capturing their interaction. Extensive experimentation demonstrates statistically significant improvement in terms of wins and ranks against 13 baselines over 28 time-series datasets.

1. INTRODUCTION

Time-series research has attracted significant interest within the data mining community, due to the fact that series data are present in a wide range of real-life domains. Time-series data often exhibit inter-class differences in terms of small sub-sequencies rather than the full series structure [17]. A recently introduced concept, named *shapelet*, represents a maximally discriminative sub-sequence of time series data. Stated more directly, shapelets identify short discriminative series segments [17, 11]. Apart from their high prediction accuracy, shapelets also offer interpretable features to domain experts. Moreover, discovering shapelets has been a hot topic in the time-series domain during the last five years [17, 11, 10, 19, 8, 12, 9].

State-of-the-art methods discover shapelets by trying a pool of candidate sub-sequences from all possible series segments [17, 10] and then sorting the top performing segments according to their target prediction qualities. Distances between series and shapelets represent shapelet-transformed [10] classification features for a series of segregation metrics, such as information gain [17, 11], F-Stat [8] or Kruskall-Wallis [9]. The brute-force candidates search approach, based on an exhaustive search of candidates, suffers from a high runtime complexity, therefore several speed-up techniques have aimed at reducing the discovery time of shapelets [11, 12, 1]. In terms of classification performance, the shapelet-transformation method constructs qualitative predictors for standard classifiers and has recently shown improvements with respect to prediction accuracy [10, 8].

This paper proposes an entirely new perspective on time-series shapelets. For the first time, we propose a mathematical formulation of the shapelet learning task as an optimization of a classification objective function. Furthermore, we propose a learning method that *learns* (not searches for) the shapelets which optimize the objective function. Concretely, we learn shapelets whose distances to series can linearly separate the time series instances by their targets, as shown in Figure 1. In comparison to existing approaches, our method can learn near-to-optimal shapelets and **true** top-K shapelet interactions. In a large pool of 28 datasets we demonstrate that the proposed method yields a large and statistically significant improvement over 13 baselines.

2. RELATED WORK

Shapelets were first proposed by [17] as time-series segments that maximally predict the target variable. All possible segments were considered as potential candidates, while the minimum distances of a candidate to all training series were used as a predictor feature for ranking the information gain accuracy of that candidate on the target variable. Other quality metrics have been proposed for evaluating the prediction accuracy of a shapelet candidate such as F-Stats [10], Kruskall-Wallis or Mood's median [8]. In addition, the minimum distance of a set of shapelets to time series can be perceived as a data transformation [10], while standard classifiers have achieved high accuracy over the shapelet-transformed representation [8].

Due to the high number of candidates, the runtime of brute-force shapelet discovery is not feasible. Therefore, a series of speed-up techniques such as early abandoning of distance computations and entropy pruning of the information gain metric have been proposed [17]. Other speed-ups rely on the reuse of computations and pruning of the search space [11], as well as exploiting projections on the SAX representation [12]. Alternatively, the training time has been reduced by elaborating the usage of infrequent shapelet candidates [7]. Moreover, hardware-based optimization have assisted the discovery of shapelets using GPUs [1]. Shapelets have been applied in a series of real-life applications. Unsupervised shapelets have also been utilized for clustering time series [19]. Shapelets have been found useful for identifying humans through their gait data [13]. Gesture recognition is another application domain that has benefited from the discovery of shapelets [5, 6]. In the domain of

KDD'14, August 24–27, 2014, New York, NY, USA.
Copyright 2014 ACM 978-1-4503-2956-9/14/08 ...$15.00.
http://dx.doi.org/10.1145/2623330.2623613 .

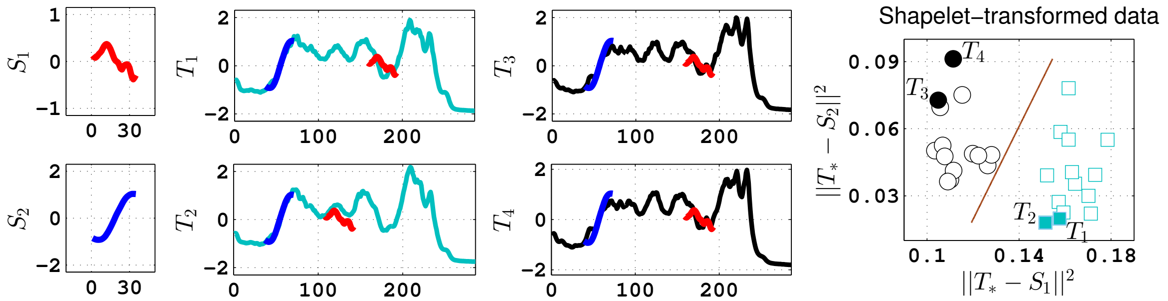

Figure 1: An illustration of two shapelets S_1, S_2 (leftmost plots) learned on the Coffee dataset. Series' distances to shapelets can optimally project the series into a 2-dimensional space, called the shapelet-transformed representation [10] (rightmost plot). The middle plots show the closest matches of the shapelets on series of two classes having light-blue and black colors.

medical and health informatics, interpretable shapelets have been used to enable efficient early classification of time-series [16, 15].

In comparison to the state-or-the-art methods, we propose a novel method that **learns** near-to-optimal shapelets directly, without the need to search exhaustively among a pool of candidates extracted from time-series segments.

3. PROPOSED METHOD

3.1 Definitions and Notations

3.1.1 Time-Series Dataset

A time-series dataset is composed of I training instances and for notation ease we assume that each series contains Q-many ordered values, even though our method can operate on variable series lengths. The dataset is defined as $T^{I \times Q}$, while the series target is a nominal variable $Y \in \{1, \ldots, C\}^I$ having C categories.

3.1.2 Sliding Window Segment

A sliding window segment of length L is an ordered sub-sequence of a series. Concretely, the segment starting at time j inside the i-th series is defined as $(T_{i,j}, \ldots, T_{i,j+L-1})$. There are totally $J := Q - L + 1$ segments in a time series provided the starting index of the sliding window is incremented by one.

3.1.3 Shapelets

A shapelet of length L is simply an ordered sequence of values from a data structure perspective. Nevertheless, shapelets semantically represent intelligence on how to discriminate the target variable of a series dataset. The K-most informative shapelets are denoted as $S \in \mathbb{R}^{K \times L}$.

3.1.4 Distances Between Shapelets and Series

The distance between the i-th series T_i and the k-th shapelet S_k is defined as the minimum distance $M_{i,k}$ (shown in Equation 1) among the distances between the shapelet S_k and each segment j of T_i [17, 18]. Informally speaking, it is the distance of a shapelet to the most similar series segment, as shown in Figure 1.

$$M_{i,k} \quad = \quad \min_{j=1,\ldots,J} \frac{1}{L} \sum_{l=1}^{L} (T_{i,j+l-1} - S_{k,l})^2 \qquad (1)$$

3.1.5 Shapelet Transformation

Minimum distances to shapelets can be characterized as a transformation of the time-series data $T \in \mathbb{R}^{I \times Q}$ into a new representation $M \in \mathbb{R}^{I \times K}$ [10]. Such a transformation reduces the dimensionality of the original time-series, because typically $K < Q$. General purpose classifiers (e.g.: SVMs, Bayesian Network, ...) have been recently shown to achieve high prediction accuracy over the new representation M [8].

3.1.6 Logistic Sigmoid Function

The logistic sigmoid function is an S shaped instance of the logistic function and is defined as $\sigma(Y) = \left(1 + e^{-Y}\right)^{-1}$. We are going to use the sigmoid function for the prediction of target variables via a logistic regression loss.

3.2 A Novel Principle

In this paper we propose a novel principle in learning time-series shapelets. Instead of searching among possible shapelet candidates from the series segments [17, 10], we propose a formal method that can directly learn optimal shapelets without needing to explore all possible candidates. Our principle can be summarized in two steps: (i) Start with rough initial guesses for the shapelets, (ii) Iteratively learn/optimize the shapelets by minimizing a classification loss function. In order to conduct the shapelet optimization, we define a novel classification model that is differentiable with respect to shapelets. Therefore, shapelets can be updated in a stochastic gradient descent optimization fashion, by taking steps towards the minimum of the classification loss function (i.e. towards maximal prediction accuracy).

3.3 Objective Function

For the sake of simplicity, the model introduced in this Section will be focused only on binary targets $Y \in \{0, 1\}^I$ and a fixed shapelet length L. A general version of the model, with extended properties, is described Section 5.

3.3.1 Learning Model

Since the minimum distances M are the new predictors in the transformed shapelets space, a linear learning model can predict approximate target values $\hat{Y} \in \mathbb{R}^{I \times K}$ via the predictors M and linear weights $W \in \mathbb{R}^K$ (plus bias $W_0 \in \mathbb{R}$), as shown in Equation 2.

$$\hat{Y}_i \quad = \quad W_0 + \sum_{k=1}^{K} M_{i,k} W_k, \qquad \forall i \in \{1, \ldots, I\} \qquad (2)$$

3.3.2 Loss Function

In this paper we are going to exploit the logistic regression classification model, because it provides an option to interpret predicted binary targets as probabilistic confidences. Such a probabilistic interpretation will ensure extending our approach to the multi-class case in Section 5. The logistic regression operates by minimizing the logistic loss, defined in Equation 3, between true targets Y and estimated ones $\hat{Y}$.

$$\mathcal{L}(Y,\hat{Y}) \quad = \quad -Y \ln \sigma(\hat{Y}) - (1-Y) \ln \left(1 - \sigma(\hat{Y})\right) \qquad (3)$$

3.3.3 Regularized Objective Function

The logistic loss function together with regularization terms represent the regularized objective function, denoted as $\mathcal{F}$ in Equation 4. ***The idea of this paper is to jointly learn the optimal shapelets S and the optimal linear hyper-plane W that minimize the classification objective $\mathcal{F}$.***

$$\operatorname*{argmin}_{S,W} \; \mathcal{F}(S,W) = \operatorname*{argmin}_{S,W} \sum_{i=1}^{I} \mathcal{L}(Y_i,\hat{Y}_i) + \lambda_W ||W||^2 \qquad (4)$$

3.4 Differentiable Soft-Minimum Function

In order to compute the derivative of the objective function, all the involved functions of the model need to be differentiable. Unfortunately, the minimum function of Equation 1 is not differentiable and the partial derivative $\frac{\partial M}{\partial S}$ is not defined. A differentiable approximation to the minimum function is introduced in this section.

For the sake of organizational clarity we will introduce the distance between the j-th segment of series i and the k-th shapelet as $D_{i,k,j}$ and define it in Equation 5.

$$D_{i,k,j} \quad := \quad \frac{1}{L} \sum_{l=1}^{L} \left(T_{i,j+l-1} - S_{k,l}\right)^2 \qquad (5)$$

A differentiable approximation of the minimum function is the popular *Soft Minimum* function that is depicted in Equation 6. A parameter α controls the *precision* of the function and the soft minimum approaches the true minimum for $\alpha \rightarrow -\infty$.

$$M_{i,k} \quad \approx \quad \hat{M}_{i,k} = \frac{\sum_{j=1}^{J} D_{i,k,j} \; e^{\alpha \, D_{i,k,j}}}{\sum_{j'=1}^{J} e^{\alpha \, D_{i,k,j'}}} \qquad (6)$$

Please note that the soft minimum has the shapelets as the only varying input, which appear embedded inside the distance definition D. We would like to explain the operating principle of the soft minimum with the aid of Figure 2.

A series from the FaceFour dataset and a shapelet are depicted in the upper plot of Figure 2. The shapelet is a slightly distorted variant of the series segment starting at time index 51. If we slide the shapelet over all the series segments and record the distance of shapelets to segments (i.e. Equation 5), then the Euclidean distances' plot in blue is achieved. Two plots in red (bottommost) illustrate the operation of the soft minimum function. Each point j of the soft minimum plots correspond to $\frac{D_{i,k,j} \; e^{\alpha \, D_{i,k,j}}}{\sum_{j'=1}^{J} e^{\alpha \, D_{i,k,j'}}}$, while the **area** under the soft-minimum plots sums up to the true minimum distance between the shapelet and the series (i.e. Equation 1). It is important to realize in the third plot ($\alpha = -20$) that the amount by which a segment distance impacts the overall minimum is directly related to how small is that segment's distance compared to other segment distances. As can be seen in the bottom plot, if $\alpha = -100$, then only the true minimum segment distance is allowed to contribute to the grand total minimum. We found out that $\alpha = -100$ is small enough to make the soft minumum yield exactly the same results as the true minimum. Therefore, we kept this value fixed throughout all our experiments.

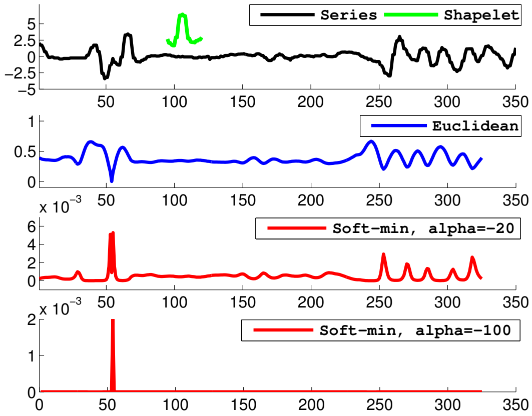

Figure 2: Illustration of the soft minimum between a shapelet (green) and all the segments of a series (black) from the FaceFour dataset

3.5 Per-Instance Objective

The optimization we will adopt in this paper is a stochastic gradient descent approach that remedies the classification error caused by one instance at a time. The Equation 7 demonstrates the decomposed objective function $\mathcal{F}_i$, which corresponds to a division of the objective of Equation 4 into per-instance losses for each time series.

$$\mathcal{F}_i \quad = \quad \mathcal{L}(Y_i,\hat{Y}_i) + \frac{\lambda_W}{I} \sum_{k=1}^{K} {W_k}^2 \qquad (7)$$

3.6 Gradients for Shapelets

The learning algorithm requires the definition of the gradients of the objective function with respect to the shapelets. The gradient of point l in shapelet k with respect to the objective of the i-th time series is defined in Equation 8 and is derived through the chain rule of derivation.

$$\frac{\partial \mathcal{F}_i}{\partial S_{k,l}} \quad = \quad \frac{\partial \mathcal{L}(Y_i,\hat{Y}_i)}{\partial \hat{Y}_i} \, \frac{\partial \hat{Y}_i}{\partial \hat{M}_{i,k}} \sum_{j=1}^{J} \frac{\partial \hat{M}_{i,k}}{\partial D_{i,k,j}} \, \frac{\partial D_{i,k,j}}{\partial S_{k,l}} \qquad (8)$$

Furthermore, the gradient of the loss with respect to the predicted target is defined in Equation 9, while the gradient of the minimum distances with respect to the estimated target is shown in Equation 10.

$$\frac{\partial \mathcal{L}(Y_i,\hat{Y}_i)}{\partial \hat{Y}_i} \quad = \quad -\left(Y_i - \sigma\left(\hat{Y}_i\right)\right) \qquad (9)$$

$$\frac{\partial \hat{Y}_i}{\partial \hat{M}_{i,k}} \quad = \quad W_k \qquad (10)$$

In addition, the gradient of the overall minimum distance with respect to a segment distance is presented in Equation 11 and the gradient of a segment distance with respect to a shapelet point is derived in Equation 12.

Algorithm 1 Learning Time-Series Shapelets

```
Require: $T \in \mathbb{R}^{I \times Q}$, Number of Shapelets $K$, Length of a shapelet $L$, Regularization $\lambda_W$, Learning Rate $\eta$, Number of iterations: maxIter
Ensure: Shapelets $S \in \mathbb{R}^{K \times L}$, Classification weights $W \in \mathbb{R}^K$, Bias $W_0 \in \mathbb{R}$
for iteration=$\mathbb{N}_1^{\text{maxIter}}$ do
  for $i = 1, \ldots, I$ do
    for $k = 1, \ldots, K$ do
      $W_k \leftarrow W_k - \eta \frac{\partial \mathcal{F}_i}{\partial W_k}$
      for $L = 1, \ldots, L$ do
        $S_{k,l} \leftarrow S_{k,l} - \eta \frac{\partial \mathcal{F}_i}{\partial S_{k,l}}$
      end for
    end for
    $W_0 \leftarrow W_0 - \eta \frac{\partial \mathcal{F}_i}{\partial W_0}$
  end for
end for
return $S, W, W_0$
```

$$\frac{\partial \hat{M}_{i,k}}{\partial D_{i,k,j}} = \frac{e^{\alpha\, D_{i,k,j}}\left(1+\alpha\left(D_{i,k,j}-\hat{M}_{i,k}\right)\right)}{\sum_{j'=1}^{J} e^{\alpha\, D_{i,k,j'}}} \tag{11}$$

$$\frac{\partial D_{i,k,j}}{\partial S_{k,l}} = \frac{2}{L}\left(S_{k,l} - T_{i,j+l-1}\right) \tag{12}$$

3.7 Gradients for Classification Weights

The hyper-plane weights W can also be learned to minimize the classification objective via stochastic gradient descent. Equation 13 shows the partial gradient of updating each weight W_k and Equation 14 presents the bias term W_0.

$$\frac{\partial \mathcal{F}_i}{\partial W_k} = -\left(Y_i - \sigma\left(\hat{Y}_i\right)\right)\hat{M}_{i,k} + \frac{2\lambda_W}{I} W_k \tag{13}$$

$$\frac{\partial \mathcal{F}_i}{\partial W_0} = -\left(Y_i - \sigma\left(\hat{Y}_i\right)\right) \tag{14}$$

3.8 Learning Algorithm

After having derived the gradients of the shapelets and the weights, we can introduce the overall learning algorithm. Our approach iterates in a series of epochs and updates the values of the shapelets and weights in the negative direction of the derivative with respect to the classification objective of each training instance.

The steps of the learning process are shown in Algorithm 1. The pseudo-code iterates over all training instances I and updates all K shapelets S and the weights W, W_0 by a learning rate η.

3.9 Convergence

The convergence of Algorithm 1 depends on two parameters, the learning rate η and the maximum number of iterations. High values for the learning rate can minimize the objective in less iterations, but pose the risk of divergence, while small learning rates require more iterations. Subsequently, the learning rate and the number of iterations should be learned via cross-validation from the training data.

For instance Figure 3 illustrates the convergence of the learning algorithm on the Coffee dataset for 57 shapelets. Both the training and the testing loss converge very smoothly for $\eta = 0.01$. As a consequence of optimizing the training loss, the train and test errors also decrease simultaneously. In addition, the regularization weight $\lambda_W = 0.001$ ensures that the train and test loss have small differences, which can be interpreted as a generalization quality without any over-fitting effect.

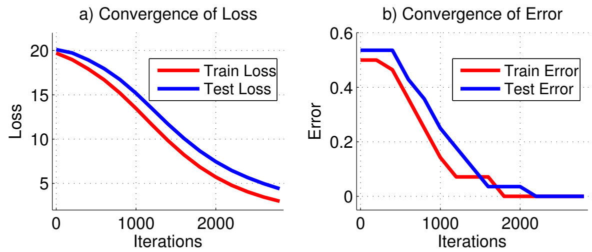

Figure 3: Convergence of the Coffee dataset, Parameters: $K = 57$, $L = 143$, $\eta = 0.01$, $\lambda_W = 0.001$, $\alpha = -100$

3.10 Model Initialization

Equation 4 is a non-convex function in terms of S and W because both are variables that need to be learned. Gradient descent techniques do not theoretically guarantee the discovery of global optima in non-linear functions. Unfortunately, non-convex optimization techniques are very slow for data mining problems, therefore gradient based approaches are often selected as a compromise between feasibility and optimality [14].

Gradient descent optimization requires a good initialization of the parameters when applied to non-convex functions. In other words, if the initialization starts the learning around a region where the global optimum is located, then the gradient can update the parameters to the exact location of the optimum.

Initialization can influence a gradient based technique significantly. We are going to illustrate the sensitivity of shapelets initialization through an experiment shown in Figure 4. For the sake of two-dimensional illustration, we initialized one shapelet (S) in the Gun-Point dataset using two values. The first 15 points of a 30 points long shapelet ($S_{1:15}$) were given a fixed initial value, while the other half points of the shapelet $S_{16:30}$ were initialized with another fixed value. Figure 4 demonstrate that different initial values of the shapelet can result in different loss values and error rates over the training instances.

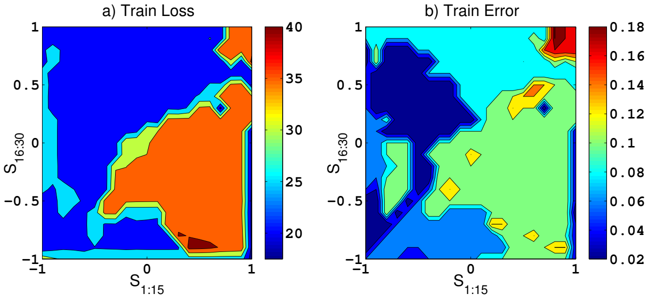

Figure 4: Sensitivity of Shapelet Initialization, Gun-Point dataset, Parameters: $L = 30$, $\eta = 0.01$, $\lambda_W = 0.01$, Iterations$= 3000$, $\alpha = -100$

In order to robustify the initialization guesses we use the K-Means centroids of all segments as initial values for the shapelets. Since centroids represent typical patterns of the data, they offer a good variety of shapes for initializing shapelets and help our method achieve high prediction accuracy. The initialization is conducted before Algorithm 1 starts, while W is also initialized randomly around 0.

3.11 Illustrating The Mechanism

An illustration of the learning algorithm is depicted in Figure 5. Two shapelets of length 40 are learned from the Gun-Point dataset. Sub-figure a) demonstrates the initialization values of the shapelets and the arrangement of the minimum values of the time series to shapelets. As can be seen, a linear hyper-plane W cannot easily separate the two classes. After 400 iterations of our method, the shapelets are updated as shown in sub-figure b). In addition, the shapelet transformed data representation M becomes almost linearly separable with few exceptions. Finally, the algorithm approaches convergence in sub-figure c) after 800 iterations. The linear hyper-plane W separates the shapelet-transformed instances of the binary dataset with just a single error (in red).

4. ANALYSIS OF OUR METHOD

4.1 Algorithmic Complexity

The baseline method which exhaustively tries candidates from series segments [17, 10] requires $\mathcal{O}(I^2Q^3)$ running time for discovering the best shapelet of a particular length Q. On the other hand, our method requires $\mathcal{O}(IQ^2 \times \text{maxIter})$, therefore our algorithm finds the best shapelet in a faster time, given that usually maxIter $<< IQ$.

4.2 Comparison to State of the Art

4.2.1 *Learning Near-To-Optimal Shapelets*

The optimal solution of Equation 4 gives the optimal shapelets, while a gradient descent approach can find a near-to-optimal minimum given an appropriate initialization.

The baseline approaches, on the other hand, provide no guarantee of optimal solutions for two primary reasons. First of all, the baselines are bound to shapelet candidates from the pool of series segments and cannot explore candidates which do not appear literally as segments. Secondly, minimizing the classification objective through candidate guesses has no guarantee of optimality, while a gradient-based optimization guarantees at least near-to-optimal minima.

4.2.2 *Capturing Interactions Among Shapelets*

The baselines find the score of each shapelet independently and then sort the individual quality of each shapelet, in order to select the top performers. However, such an approach does not take into account interactions among patterns. In other words, two shapelets can be individually sub-optimal, but when combined together they can improve the results. In fact, this problem is well known in data mining and referred to as variable subset selection [2].

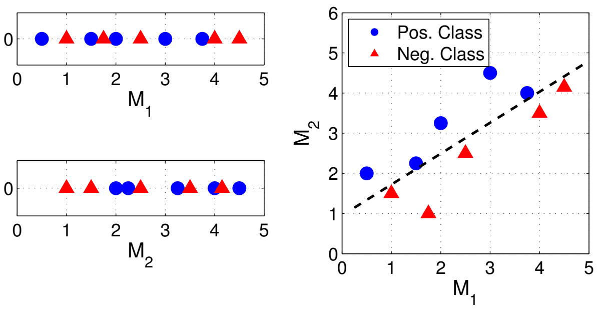

Figure 6: Interactions among Shapelets Enable Individually Unsuccessful Shapelets (left plots) to Excel in Cooperation (right plot)

For instance, Figure 6 demonstrates an example on how interactions among shapelets can become a game changing factor. On the left plots, we show the minimum distances of series to two shapelets. As can be observed, the individual discriminative quality of the shapelets is poor. On the other hand, a simple 2-dimensional interaction of exactly the **same** distances M_1, M_2 can yield drastically improved results, as shown on the right plot. When combined together, the distances to those shapelets can create a linearly separable discrimination, i.e. a perfect classification accuracy.

If the baseline's exhaustive discovery approach would attempt to select the **true** top-K interaction of shapelets out of $I(Q-L+1)$ candidates, then it will need to check the interaction of: $\begin{pmatrix} I(Q-L+1) \\ K \end{pmatrix} = \frac{(I(Q-L+1))!}{K!(I(Q-L+1)-K)!}$ many combinations of candidates. For instance finding the **true** top 100 shapelets of length 30 from the Adiac dataset with $I = 390$ and $Q = 176$ requires checking 3.42×10^{317} combination trials using the baseline's approach. Clearly, the exhaustive search baseline cannot find **true** top-K shapelet interactions within a feasible time-frame. On the contrary, our method can find the interactions at a simple linear scale K, due to the property of jointly learning the shapelets and their interactions.

4.2.3 *Weaker Aspects of Our Paper*

Our method relies on more hyper-parameters than the baselines, such as the learning rate η, the number of iterations, the regularization parameter λ_W and the soft-min precision α. However, given the high prediction accuracy that will be demonstrated in Section 6, we argue that the very high accuracy by far out-weights the model's learning efforts.

The time needed for the baselines to compute the top-K shapelets is not significantly large with respect to the time needed to find a single shapelet. Such a behavior comes from the fact that the quality of each candidate is known and ready in the end of the discovery. On the other hand, our method needs K-many time units for K shapelets, w.r.t. learning one shapelet. However, such a disadvantage in time for large K is well spent in terms of accuracy, because our method can learn **true** top-K shapelet interactions and significantly improve the classification accuracy. Moreover, we believe that our method may yield to further improvements in efficiency by exploiting "early abandoning" and caching of partial results (as in [7]). For brevity we ignore such issues here and focus on forcefully demonstrating the improvements in accuracy.

5. LEARNING GENERAL SHAPELETS

The model presented in Section 3.3 can be generalized to multi-class labels and multi-size shapelets. Basically the model is extended to classify multi-class targets and capture interactions among shapelets of various sizes.

5.1 Decomposition of the Multi-Class Problem Into One-vs-all Subproblems

In order to learn from multi-class targets $Y \in \{1, \ldots, C\}^I$ with C categories, we will convert the problem into C-many one-vs-all sub-problems. Each sub-problem will discriminate one class against all the others. The one-vs-all binary targets $Y^b \in \{0,1\}^{I \times C}$ are defined in Equation 15.

$$Y^b_{i,c} = \begin{cases} 1 & Y_i = c \\ 0 & Y_i \neq c \end{cases}, \quad \forall i \in \{1, \ldots, N\}, \quad \forall c \in \{1, \ldots, C\} \quad (15)$$

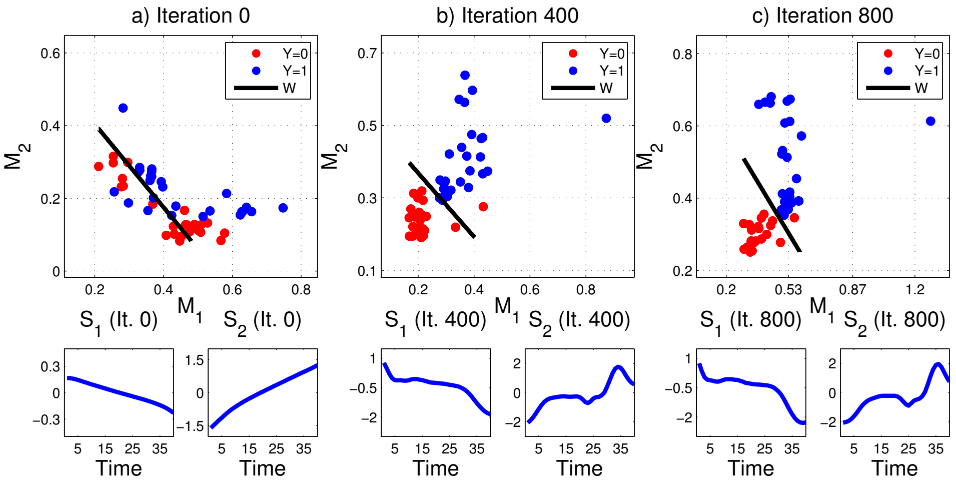

Figure 5: Learning Two Shapelets on the Gun-Point Dataset: Parameters $L = 40, \eta = 0.01, \lambda_W = 0.01, \alpha = -100$

In fact, the conversion to one-vs-all sub-problems will be useful for the operation of the logistic regression classifier. The output of the logistic regression for a binary problem can be perceived as a confidence probability. Therefore, the index of the most confident among the C-many classifiers is selected as the predicted categorical value of a test instance.

5.2 Interactions among Shapelets having Various Lengths

Capturing interactions among shapelets having various lengths is another aspect of the extended method. Our generalized model learns R different scales of shapelet lengths starting at a minimum L^{min} as $\{L^{\text{min}}, 2L^{\text{min}}, \ldots, RL^{\text{min}}\}$. The shapelets therefore will be defined as $S \in \mathbb{R}^{R \times K \times *}$, where $S \in \bigcup_{r=1}^{R} \mathbb{R}^{K \times rL^{\text{min}}}$, and represent K-many shapelets for each scale R, i.e. totally KR shapelets. The length of a shapelet at scale $r \in \{1, \ldots, R\}$ is $r \cdot L^{min}$. Consequently, the number of segments in a time series depends on the scale of the shapelet's length to be matched against and is $J(r) = Q - r \cdot L^{min} + 1$.

5.3 Generalized Objective Function

The objective function of the generalized model is presented in Equation 16, which is a regularized logistic regression loss between the true targets and the predicted ones shown in Equation 17. The notation $M_{r,i,k}$ identifies the minimum distance of the i-th series to the k-th shapelet of scale r, i.e. to $S_{r,k} \in \mathbb{R}^{r \cdot L^{min}}$. In addition, the weight $W_{c,r,k}$ identifies the class c classifier and the weight of the k-th shapelet at scale r.

$$\underset{S,W}{\text{argmin}} \quad \mathcal{F} \quad = \quad \sum_{i=1}^{I} \sum_{c=1}^{C} \mathcal{L}(Y_{i,c}^{b}, \hat{Y}_{i,c}^{b}) + \lambda_W ||W||^2 \quad (16)$$

$$\hat{Y}_{i,c}^{b} \quad = \quad W_{c,0} + \sum_{r=1}^{R} \sum_{k=1}^{K} M_{r,i,k} W_{c,r,k} \quad (17)$$

5.4 Classification of Test Instances

Once the model is learned, a test instance indexed t is classified as the one-vs-all classifier which yields maximum confidence, as presented in Equation 18. The algorithmic complexity of classifying a test instance is $O(CRKJ(\cdot))$, but since C, R, K are asymptotically smaller values than $J(\cdot)$, the big-O notation complexity is $O(J(\cdot))$.

$$\hat{Y}_{\mathbf{t}} \quad \leftarrow \quad \underset{c \in \{1,\ldots,C\}}{\text{argmax}} \; \sigma\left(\hat{Y}_{\mathbf{t},c}^{b}\right), \qquad \forall t \in \{1, \ldots, I^{Test}\} \quad (18)$$

5.5 Generalized Soft-Minimum

The soft minimum function can be trivially generalized to include the notation for the scales r as shown in Equation 19. The distance between the k-th shapelet at scale r and the j-th segment of time series i is denoted as $D_{r,i,k,j}$ in Equation 20.

$$M_{r,i,k} \quad \approx \quad \hat{M}_{r,i,k} = \frac{\sum_{j=1}^{J(r)} D_{r,i,k,j} \, e^{\alpha \, D_{r,i,k,j}}}{\sum_{j'=1}^{J(r)} e^{\alpha \, D_{r,i,k,j'}}} \quad (19)$$

$$D_{r,i,k,j} \quad = \quad \frac{1}{r \cdot L^{\text{min}}} \sum_{l=1}^{r \cdot L^{\text{min}}} \left(T_{i,j+l-1} - S_{r,k,l}\right)^2 \quad (20)$$

5.6 Gradients of Generalized Objective Function

The objective function can be split per each instance i and the loss of each one-vs-all classifier c and denoted in Equation 21 as $F_{i,c}$.

$$\mathcal{F}_{i,c} \quad = \quad \mathcal{L}(Y_{i,c}^{b}, \hat{Y}_{i,c}^{b}) + \frac{\lambda_W}{IC} \sum_{r=1}^{R} \sum_{k=1}^{K} {W_{c,r,k}}^2 \quad (21)$$

5.6.1 *Shapelet Gradients*

The derivative of the per-cell objective $F_{i,c}$ with respect to each shapelet $S_{r,k,l}$ is shown in Equation 22.

$$\frac{\partial \mathcal{F}_{i,c}}{\partial S_{r,k,l}} = -\left(Y_{i,c}^{b} - \sigma\left(\hat{Y}_{i,c}^{b}\right)\right) \frac{\partial \hat{M}_{r,i,k}}{\partial S_{r,k,l}} W_{c,r,k} \quad (22)$$

Moreover, the derivative of the minimum distances with respect to the generalized shapelets is defined in Equations 23-24.

$$\frac{\partial \hat{M}_{r,i,k}}{\partial D_{r,i,k,j}} = \frac{e^{\alpha\, D_{r,i,k,j}} \left(1 + \alpha \left(D_{r,i,k,j} - \hat{M}_{r,i,k}\right)\right)}{\sum_{j'=1}^{J(r)} e^{\alpha\, D_{r,i,k,j'}}} \quad (23)$$

$$\frac{\partial D_{r,i,k,j}}{\partial S_{r,k,l}} = \frac{2\left(S_{r,k,l} - T_{i,j+l-1}\right)}{r \cdot L^{\text{min}}} \quad (24)$$

5.6.2 *Classification Weights' Gradients*

The gradients of the per-cell objective with respect to the generalized weights and the bias terms are presented in Equations 25-26.

$$\frac{\partial \mathcal{F}_{i,c}}{\partial W_{c,r,k}} = -\left(Y_{i,c}^b - \sigma\left(\hat{Y}_{i,c}^b\right)\right) \hat{M}_{r,i,k} + \frac{\lambda_W W_{c,r,k}}{IC} \quad (25)$$

$$\frac{\partial \mathcal{F}_{i,c}}{\partial W_{c,0}} = -\left(Y_{i,c}^b - \sigma\left(\hat{Y}_{i,c}^b\right)\right) \quad (26)$$

5.6.3 *Optimized Learning Algorithm*

Algorithm 2 summarizes all the steps of the learning process. The first section of the procedure pre-computes terms which are used frequently in the gradients of the shapelets, such as ξ, D, ψ, ϑ. The pre-computations boost the learning time and avoid computing the same terms repeatedly. The second part of the algorithm updates the weights and the shapelets using the defined gradients and the precomputed terms.

6. EXPERIMENTAL RESULTS

6.1 Setup And Reproducibility

6.1.1 *Datasets*

For the sake of equivalent comparison, we selected exactly the same set of datasets as the closest baselines [10, 8]. A large pool of 28 datasets consisting of time-series datasets having various numbers of instances, lengths and number of classes is selected and details are shown in Table 1. In order to ensure a fair comparison with the baselines, we used the default train and test data splits, **same as the baselines** [10, 8]. The datasets are available through the UCR[1] and UEA[2] websites.

6.1.2 *Reproducibility and Hyper-parameter Search*

Our method (hereafter denoted as LTS, for **L**earning **T**ime-**S**eries **S**hapelets) requires the tuning of a series of hyper-parameters, which were found through a grid search approach using cross-validation over the training data. The number of shapelets was searched in a range of $K \in \{0.05, 0.15, 0.3\}$, which is a fraction of the series length, e.g. $K = 0.3$ means 30% of Q. Similarly, $L^{\text{min}} \in \{0.025, 0.075, 0.125, 0.175, 0.2\} \times 100\%$ of Q, while three scales of shapelet lengths were searched from $R \in \{1, 2, 3\}$. The regularization parameter was one of $\lambda_W \in \{0.01, 0.1, 1\}$. The learning rate was kept fixed at a small value of $\eta = 0.01$, while the number of iterations is selected from maxIter$\in \{2000, 5000, 10000\}$. All our method' parameters for all datasets are shown in Table 1. **The authors are devoted to promote reproducibility, therefore the source code, datasets and instructions are made publicly available**[3].

Algorithm 2 Generalized Shapelets Learning

```
Require: Time series $T \in \mathbb{R}^{I \times Q}$, Binary labels $Y^b \in \mathbb{R}^{I \times C}$,
    Number of shapelets $K$, Learn Rate $\eta$, Regularization $\lambda_W$,
    Scales of shapelet lengths $R \in \mathbb{N}$, Minimum Shapelet Length
    $L^{\text{min}}$, Number of Iterations: maxIter
Ensure: Shapelets $S \in \mathbb{R}^{R \times K \times *}$, Classification weights $W \in$
    $\mathbb{R}^{R \times C \times K}$, $W_0 \in \mathbb{R}^C$
Initialize $S, W, W_0$
for iteration=$\{1, \ldots,$ maxIter$\}$ do
  for $i = \{1, \ldots, I\}$ do
    {Pre-compute Terms}
    for $r = \{1, \ldots, R\},\ k = \{1, \ldots, K\}$ do
      for $j = \{1, \ldots, J(r)\}$ do
        $D_{r,i,k,j} := \frac{1}{r \cdot L^{\text{min}}} \sum_{l=1}^{r \cdot L^{\text{min}}} \left(T_{i,j+l-1} - S_{r,k,l}\right)^2$
        $\xi_{r,i,k,j} := e^{\alpha\, D_{r,i,k,j}}$
      end for
      $\psi_{r,i,k} := \sum_{j=1}^{J(r)} \xi_{r,i,k,j}$
      $\hat{M}_{r,i,k} := \frac{1}{\psi_{r,i,k}} \sum_{j=1}^{J(r)} D_{r,i,k,j}\, \xi_{r,i,k,j}$
    end for
    for $c = \{1, \ldots, C\}$ do
      $\sigma(\hat{Y}_{i,c}^b) := \left(1 + e^{-\sum_{r=1}^{R} \sum_{k=1}^{K} \hat{M}_{r,i,k} W_{c,r,k}}\right)^{-1}$
      $\vartheta_{i,c} := Y_{i,c}^b - \sigma(\hat{Y}_{i,c}^b)$
    end for
    {Learn Shapelets and Classification Weights}
    for $c = \{1, \ldots, C\}$ do
      for $r = \{1, \ldots, R\},\ k = \{1, \ldots, K\}$ do
        $\mathbf{W}_{c,r,k} \mathrel{+}= \eta \left(\vartheta_{i,c} \hat{M}_{r,i,k} - \frac{2\lambda_W}{IC} \mathbf{W}_{c,r,k}\right)$
        for $j = \{1, \ldots, J(r)\}$ do
          $\phi_{r,i,k,j} := \frac{2\xi_{r,i,k,j}\left(1+\alpha\left(D_{r,i,k,j} - \hat{M}_{r,i,k}\right)\right)}{r \cdot L^{\text{min}} \psi_{r,i,k}}$
          for $l = \{1, \ldots, r \cdot L^{\text{min}}\}$ do
            $\mathbf{S}_{r,k,l} \mathrel{+}= \eta\, \vartheta_{i,c}\, \phi_{r,i,k,j} \times$
                $\left(\mathbf{S}_{r,k,l} - T_{i,j+l-1}\right) W_{c,r,k}$
          end for
        end for
      end for
      $\mathbf{W}_{c,0} \mathrel{+}= \eta\, \vartheta_{i,c}$
    end for
  end for
end for
return $S, W, W_0$
```

6.1.3 *Baselines*

Thirteen different baselines were compared against, which are grouped into the following four clusters.

- **Shapelet Tree Methods**, constructed from shapelets whose qualities are measured using: *i)* Information gain quality criterion (IG) [17, 11], *ii)* Kruskall-Wallis quality criterion (KW) [8], *iii)* F-Stats quality criterion (FST) [10] and *iv)* the Mood's Median Criterion (MM) [8].

- **Basic Classifiers** [10], learned over shapelet-transformed data, such as: Nearest Neighbors (1NN), Naive Bayes (NB) and C4.5 tree (C4.5).

- **More Complex Classifiers** [8], learned over shapelet transformed data, such as: Bayesian Networks (BN), Random

[1] `http://www.cs.ucr.edu/~eamonn/time_series_data/`

[2] `http://www.uea.ac.uk/computing/machine-learning/shapelets/shapelet-data`

[3] `http://fs.ismll.de/publicspace/LearningShapelets/`

Table 1: Dataset Information, Parameter Search Results and Running Time for The Best Shapelet [8]

Dataset Information				**Parameter Values (LTS)**					**Run-Time (Best Shap.)**	
Dataset	Train/Test	Length	Cls.	K	$L^{\min}$	R	λ_W	maxIter	F-Stat (Sec)	LTS (Sec)
Adiac	390/391	176	37	0.3	0.2	3	0.01	10000	4509.91	3017.23
Beef	30/30	470	5	0.15	0.125	3	0.01	10000	1251.21	293.68
Beetle/Fly	20/20	512	2	0.15	0.125	1	0.01	5000	21496.51	131.015
Bird/Chicken	20/20	512	2	0.3	0.075	1	0.1	10000	20465.63	81.405
Chlorine.	467/3840	166	3	0.3	0.2	3	0.01	10000	15681.39	558.51
Coffee	28/28	286	2	0.05	0.075	2	0.01	5000	258.15	90.96
Diatom.	16/306	345	4	0.3	0.175	2	0.01	10000	53.91	173.1
DP_Little	400/645	250	3	0.15	0.175	1	1	5000	78005.7	1525.595
DP_Middle	400/645	250	3	0.3	0.025	3	1	10000	91208.52	910.33
DP_Thumb	400/645	250	3	0.05	0.175	3	0.1	5000	123766.49	963.765
ECGFiveDays	23/861	136	2	0.05	0.125	2	0.01	5000	149.1	29.365
FaceFour	24/88	350	4	0.3	0.175	3	1	5000	4556.41	386.45
GunPoint	50/150	150	2	0.15	0.2	3	0.1	10000	569.42	46.69
ItalyPower.	67/1029	24	2	0.3	0.2	3	0.01	5000	1.75	10.285
Lighting7	70/73	319	7	0.05	0.075	3	1	5000	14912.74	394.44
MedicalImages	381/760	99	10	0.3	0.2	2	1	10000	7742.97	406.725
MoteStrain	20/1252	84	2	0.3	0.2	3	1	10000	10.76	16.875
MP_Little	400/645	250	3	0.3	0.2	3	0.01	5000	88071.5	965.27
MP_Middle	400/645	250	3	0.05	0.2	2	0.01	5000	134731.54	940.555
Otoliths	64/64	512	2	0.15	0.125	3	0.01	2000	55874.19	407.835
PP_Little	400/645	250	3	0.15	0.125	3	1	10000	79993.31	890.925
PP_Middle	400/645	250	3	0.15	0.175	2	0.01	10000	57815.02	1574.805
PP_Thumb	400/645	250	3	0.3	0.175	2	0.1	10000	91401.49	1449.36
SonyAIBO.	20/601	70	2	0.3	0.125	2	0.01	10000	6.73	11.415
Symbols	25/995	398	6	0.05	0.175	1	0.1	5000	8901.28	308.99
SyntheticControl	300/300	60	6	0.15	0.125	3	0.01	5000	984.36	219.97
Trace	100/100	275	4	0.15	0.125	2	0.1	10000	54128.53	275.375
TwoLeadECG	23/1139	82	2	0.3	0.075	1	0.1	10000	3.12	15.415

Forest (RAF), Rotation Forest (ROF) and Support Vector Machines (SVM).

- **Other Related Methods:** The Fast Shapelets (FSH) [12] exploits a fast random projection technique on the SAX representation, while the **Dynamic Time Warping** (DTW) classifier on the raw time-series data is also selected due to its reputation as a strong similarity metric [4].

6.2 Very High Prediction Accuracy

We compared our method of learning shapelets (denoted as LTS) against the selected baselines in terms of classification accuracy ratio (fraction of correct classifications) as shown in Table 2. The best method per dataset is highlighted in **bold**.

Our method LTS has a very large superiority in terms of Absolute Wins (17.28 absolute wins in 28 datasets against 13 baselines) and 1-to-1 wins, as indicated by the respected rows in the end of the table. Each dataset awards one point, which is split into fractions in case of draws. In addition, we compared the ranks of the classifiers and found out that LTS has a significantly better rank of 1.946 ± 0.536 against the closest baseline's (SVM) rank 4.554 ± 1.180. In order to bullet-proof our claim, we ran the well-known Wilcoxon signed ranks test [3] against all baselines and found out that all results are statistically significant at $p < 0.05$, as can be deduced by the p-values of the most bottom row.

6.3 Competitive Running Time

Since the idea of this paper is entirely novel, our first priority is to evaluate its prediction accuracy rather than elaborating on speed-up techniques, as in the fast shapelets approach [12]. Nevertheless, we would like to show that our method is indeed feasible and competitive in terms of running time and faster than the exhaustive candidate search approach [10, 8]. We compared the time needed to find the best shapelet of each dataset against the F-Stat metric, which is the fastest quality metric [10, 8]. The best shapelet runtime comparison is advocated by our baseline [8], in order to ensure that methods can process the same number of candidates. As can be seen from Table 1, our method can learn the shapelet within a faster time (57 times faster in average) compared to the baseline, which is an indication that our method is practically feasible in terms of running time. Each execution of our method searched over five different shapelet sizes $\{0.025, 0.075, 0.125, 0.175, 0.2\} \times Q$ and the other parameters were set to $\eta = 0.01$, maxIter$= 3000$ and $\lambda_W = 0.001$.

7. CONCLUSION

In this study we introduced a novel perspective into learning time-series shapelets. In contrast to related work which searches for top shapelets from a pool of candidates, we propose a novel mathematical formulation of the task via a classification objective function. In addition, we introduced a learning algorithm which *learns* near-to-optimal shapelets by exploring shapelet interactions. An extensive experimentation on 28 time-series datasets and 13 base-

Table 2: Accuracy Ratios involving 28 Time Series Datasets and 13 Baselines

#	Dataset	IG	KW	FST	MM	DTW	C4.5	1NN	NB	BN	RAF	ROF	SVM	FSH	LTS
1	Adiac	0.299	0.266	0.156	0.271	0.491	0.243	0.253	0.281	0.251	0.304	0.307	0.238	**0.558**	0.542
2	Beef	0.500	0.333	0.567	0.300	0.433	0.600	0.833	0.733	**0.900**	0.600	0.700	0.867	0.517	0.800
3	Beetle/Fly	0.775	0.700	0.900	0.800	0.650	0.750	**1.000**	0.925	0.975	0.900	0.950	0.975	0.742	0.950
4	Bird/Chicken	0.850	0.875	0.900	0.875	0.725	0.900	0.975	0.875	0.950	0.950	0.925	0.950	0.843	**1.000**
5	ChlorineConcentration	0.588	0.520	0.535	0.521	0.634	0.565	0.569	0.460	0.571	0.576	0.635	0.562	0.578	**0.743**
6	Coffee	0.964	0.857	**1.000**	0.857	0.464	0.857	**1.000**	0.929	0.964	**1.000**	0.893	**1.000**	0.925	**1.000**
7	DiatomSizeReduction	0.722	0.621	0.765	0.448	0.925	0.752	0.935	0.788	0.902	0.804	0.830	0.922	0.869	**0.951**
8	DP_Little	0.654	0.680	0.603	0.710	0.493	0.659	0.728	0.735	0.729	0.730	0.747	**0.752**	0.566	0.727
9	DP_Middle	0.705	0.693	0.619	0.737	0.546	0.712	0.737	0.740	0.747	0.755	0.768	**0.796**	0.581	0.758
10	DP_Thumb	0.581	0.720	0.560	0.703	0.530	0.580	0.607	0.630	0.639	0.641	0.671	0.698	0.580	**0.740**
11	ECGFiveDays	0.775	0.872	0.990	0.928	0.828	0.962	0.984	0.964	0.995	0.933	0.986	0.990	0.996	**1.000**
12	FaceFour	0.841	0.443	0.750	0.409	0.830	0.761	**1.000**	0.977	**1.000**	0.875	0.989	0.977	0.909	**1.000**
13	GunPoint	0.893	0.940	0.953	0.920	0.913	0.907	0.980	0.920	0.993	0.960	0.987	**1.000**	0.932	**1.000**
14	ItalyPowerDemand	0.892	0.910	0.931	0.911	0.961	0.910	0.921	0.925	0.924	0.930	0.920	0.921	0.921	**0.962**
15	Lighting7	0.493	0.480	0.411	0.274	0.726	0.534	0.493	0.575	0.658	0.644	0.658	0.699	0.601	**0.877**
16	MedicalImages	0.488	0.471	0.508	0.490	0.690	0.449	0.457	0.174	0.282	0.508	0.515	0.525	0.608	**0.734**
17	MoteStrain	0.825	0.840	0.840	0.840	0.816	0.844	0.903	0.888	0.891	0.846	0.870	0.887	0.785	**0.913**
18	MP_Little	0.664	0.697	0.578	0.703	0.558	0.634	0.685	0.688	0.695	0.714	0.752	0.750	0.565	**0.758**
19	MP_Middle	0.710	0.750	0.609	0.720	0.470	0.733	0.709	0.720	0.711	0.752	0.747	0.769	0.605	**0.780**
20	Otoliths	0.672	0.609	0.578	0.547	0.594	0.656	0.719	0.688	0.641	0.656	0.594	0.641	0.560	**0.766**
21	PP_Little	0.596	0.720	0.586	0.673	0.495	0.574	0.672	0.692	0.701	0.666	0.698	**0.721**	0.549	0.710
22	PP_Middle	0.614	0.683	0.581	0.697	0.499	0.625	0.685	0.698	0.714	0.705	0.754	0.759	0.580	**0.767**
23	PP_Thumb	0.608	0.713	0.591	0.730	0.526	0.595	0.677	0.694	0.695	0.678	0.728	**0.755**	0.536	0.715
24	SonyAIBORobotSurface	0.845	0.727	**0.953**	0.749	0.699	0.845	0.840	0.790	0.897	0.852	0.890	0.867	0.698	0.952
25	Symbols	0.780	0.557	0.801	0.574	0.934	0.471	0.856	0.780	0.923	0.846	0.844	0.846	0.930	**0.959**
26	SyntheticControl	0.943	0.900	0.957	0.857	0.973	0.903	0.930	0.780	0.767	0.890	0.920	0.873	0.917	**1.000**
27	Trace	0.980	0.940	0.980	**1.000**	0.990	0.980	0.980	0.980	**1.000**	0.980	0.980	0.980	**1.000**	**1.000**
28	TwoLeadECG	0.851	0.764	0.970	0.853	0.795	0.853	0.995	0.991	0.988	0.961	0.980	0.993	0.922	**1.000**
Absolute Wins		**0.00**	**0.00**	**1.20**	**0.25**	**0.00**	**0.00**	**1.53**	**0.00**	**1.58**	**0.20**	**0.00**	**4.70**	**1.25**	**17.28**
LTS 1-to-1 Wins		**28**	**27**	**26**	**26**	**28**	**28**	**23**	**27**	**23**	**26**	**24**	**20**	**26**	**-**
LTS 1-to-1 Draws		**0**	**0**	**1**	**1**	**0**	**0**	**2**	**0**	**2**	**1**	**1**	**2**	**1**	**-**
LTS 1-to-1 Losses		**0**	**1**	**1**	**1**	**0**	**0**	**3**	**1**	**3**	**1**	**3**	**6**	**1**	**-**
Rank Mean		**9.768**	**9.982**	**9.107**	**9.500**	**9.804**	**10.036**	**6.196**	**7.714**	**5.518**	**6.321**	**5.357**	**4.554**	**9.196**	**1.946**
Rank Confidence Interval		**1.016**	**1.259**	**1.318**	**1.273**	**1.867**	**0.781**	**1.195**	**1.091**	**1.150**	**0.743**	**0.898**	**1.180**	**1.519**	**0.536**
Rank Standard Deviation		**2.743**	**3.398**	**3.559**	**3.436**	**5.040**	**2.108**	**3.228**	**2.944**	**3.104**	**2.005**	**2.423**	**3.186**	**4.102**	**1.448**
Wilcoxon Test p-values		**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.003**	**0.000**	**-**

lines is conducted. Our method outperforms all the baselines with statistically significant margins in terms of both wins and ranks.

8. ACKNOWLEDGMENT

This study was partially co-funded by the Seventh Framework Programme (FP7) of the European Commission, through project REDUCTION[4] (#288254).

9. REFERENCES

[1] K.-W. Chang, B. Deka, W. mei W. Hwu, and D. Roth. Efficient pattern-based time series classification on gpu. In M. J. Zaki, A. Siebes, J. X. Yu, B. Goethals, G. I. Webb, and X. Wu, editors, *ICDM*, pages 131–140. IEEE Computer Society, 2012.

[2] A. Das and D. Kempe. Algorithms for subset selection in linear regression. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing*, STOC '08, pages 45–54, New York, NY, USA, 2008. ACM.

[3] J. Demšar. Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.*, 7:1–30, Dec. 2006.

[4] 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. *PVLDB*, 1(2):1542–1552, 2008.

[5] B. Hartmann and N. Link. Gesture recognition with inertial sensors and optimized dtw prototypes. In *IEEE International Conference on Systems Man and Cybernetics*, 2010.

[6] B. Hartmann, I. Schwab, and N. Link. Prototype optimization for temporarily and spatially distorted time series. In *the AAAI Spring Symposia*, 2010.

[7] Q. He, F. Zhuang, T. Shang, Z. Shi, et al. Fast time series classification based on infrequent shapelets. In *11th IEEE International Conference on Machine Learning and Applications*, 2012.

[8] J. Hills, J. Lines, E. Baranauskas, J. Mapp, and A. Bagnall. Classification of time series by shapelet transformation. *Data Mining and Knowledge Discovery*, 2013.

[9] J. Lines and A. Bagnall. Alternative quality measures for time series shapelets. In *Intelligent Data Engineering and Automated Learning*, volume 7435 of *Lecture Notes in Computer Science*, pages 475–483. 2012.

[10] J. Lines, L. Davis, J. Hills, and A. Bagnall. A shapelet transform for time series classification. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2012.

[11] A. Mueen, E. 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*, 2011.

[12] T. Rakthanmanon and E. Keogh. Fast shapelets: A scalable algorithm for discovering time series shapelets. *Proceedings of the 13th SIAM International Conference on Data Mining*, 2013.

[13] P. Sivakumar and T. Shajina. Human gait recognition and classification using time series shapelets. In *IEEE International Conference on Advances in Computing and Communications*, 2012.

[14] E. W. Wild. *Optimization-based Machine Learning and Data Mining*. ProQuest, 2008.

[15] Z. Xing, J. Pei, and P. Yu. Early classification on time series. *Knowledge and information systems*, 31(1):105–127, 2012.

[16] Z. Xing, J. Pei, P. Yu, and K. Wang. Extracting interpretable features for early classification on time series. *Proceedings of the 11th SIAM International Conference on Data Mining*, 2011.

[17] L. Ye and E. Keogh. Time series shapelets: a new primitive for data mining. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2009.

[18] L. Ye and E. Keogh. Time series shapelets: a novel technique that allows accurate, interpretable and fast classification. *Data Mining and Knowledge Discovery*, 22(1):149–182, 2011.

[19] J. Zakaria, A. Mueen, and E. Keogh. Clustering time series using unsupervised-shapelets. In *Proceedings of the 12th IEEE International Conference on Data Mining*, 2012.

[4] `www.reduction-project.eu`