SE4TeC: A Scalable Engine For efficient and expressive Time Series Classification

Jingwei Zuo
DAVID Lab, University of Versailles
University of Paris-Saclay
Versailles, France
jingwei.zuo@uvsq.fr

Karine Zeitouni
DAVID Lab, University of Versailles
University of Paris-Saclay
Versailles, France
karine.zeitouni@uvsq.fr

Yehia Taher
DAVID Lab, University of Versailles
University of Paris-Saclay
Versailles, France
yehia.taher@uvsq.fr

***Abstract*—Time Series (TS) data are ubiquitous in enormous application fields, such as medicine, multimedia, and finance. In this paper, we present SE4TeC: A Scalable Engine for efficient and expressive Time Series Classification, which brings novel optimizations to the state-of-the-art shapelet-based algorithm: (i) More efficient feature extraction; (ii) Better interpretability of both feature extraction and classification process; (iii) Scalability in the context of Big Data. SE2TeC's effectiveness and efficiency are experimentally demonstrated over real-life datasets.**

I. Introduction

Tony is informed unhealthy heart status on the basis of his electrocardiogram (ECG) during a medical diagnosis. An experienced doctor can easily correlate the abnormal ECG with the diseases, then explain to Tony the symbolic abnormality in the ECG and the relevant treatment. Nowadays, Machine Learning technique can partly replace the role of an experienced doctor and do the diagnosis very accurately. In Tony's case, the electrical activity of the heart from physiological sensor is collected as Time Series (**TS**). From the perspective of the machine, the diagnosis can be considered as a Time Series classification (**TSC**) problem.

The classical approaches [1], [2] on TSC problems are usually based on the statistical features extracted from time series, such as mean, standard deviation of subsequences, which are assumed to represent the global characteristics of time series. Intuitively, they get very superficial information with low noise tolerance. On the basis of solving these limitations, **Shapelet** [3] has attracted great interest over the past years, owing to its high discriminative feature and good interpretability. However, extracting shapelets from data series is accomplished with large computation cost. Even for small sized datasets, the algorithm can take days. This is mainly due to the repeated similarity search between a sub-sequence (i.e. a candidate shapelet) and TS instances in the database. Some typical speed-up techniques (i.e., indexing [4], lower-bounding [5] and early abandoning [3]), introduce always extra parameters, which is difficult to operate without prior knowledge. Some low-dimensional representation methods have also been proposed, such as Piece-wise Aggregate Approximation (PAA) [6], which computes the mean of each subsequence of time series in a given length, and transforms the raw data in coarse-grained sub-components. Symbolic Aggregate approXimation (SAX) [7], transforms subsequences of raw time series into value-characterized symbols, which is eligible for a hierarchic indexing iSAX [8] to accelerate similarity search. Fast Shapelet [9] proves its efficiency even scalability based on SAX and random projection. However, a loss of feature information is unavoidable after the reduction of dimensions, which requires a trade-off between efficiency and accuracy.

Our work is biased towards raw time series processing which has a higher accuracy performance, but a relatively high time complexity [3], [10]. Unlike some hardware-based implementations, such as using GPUs to accelerate the similarity calculation [11], we focus here on the scalability of TSC based on shapelet extraction. Traditional TSC algorithms on raw TS data are not applicable for big data context, because of their low scalability. Here are our contributions in this paper:

1) We propose a novel method to evaluate the shapelet in batches
2) We introduce a scalable engine to extract the shapelet expressively
3) Based on the scalable engine, we propose an acceleration strategy to extract the shapelet more efficiently

The rest of this paper is organized as follows. In Section 2, we review the background and state the research problems. We present our scalable engine for Time Series Classification in Section 3. Section 4 shows an empirical evaluation and performance comparison with rival solutions. Finally, we give our conclusions and perspectives for future work in Section 5.

II. Background

A. Definition and notation

We start with defining shapelets and the other notions used in the paper.

Definition 1: *A Time Series T* is a sequence of real-valued numbers T=$(t_1, t_2, ..., t_i, ..., t_n)$, where n is the length of T.

Definition 2: *A subsequence $T_{i,m}$* of T is a continuous subset of values from T with length m starting from position i. $T_{i,m} = (t_i, t_{i+1}, ..., t_{i+m-1})$, where $i \in [0, n-m+1]$.

Definition 3: *Shapelet $\hat{s}$* is a time series subsequence which shape is particularly representative of a class. As such, it shows an interpretable feature which can distinguish one class from the others.

Definition 4: *A Dataset D* is a collection of time series T_i, and its class label c_i, Formally, D = $<T_1, c_{j_1}>$,$<T_2, c_{j_2}>$,...,$<T_N, c_{j_N}>$, where N is the number of instances in D. $C = c_1, c_2, ..., c_{|C|}$ is a collection of class labels, where $|C|$ denotes the number of labels.

Definition 5: *Z-Normalization Time Series* is a formal representation of Time Series, which is defined as $ZNormal(T) = \frac{T-\mu}{\sigma}$, where μ is the sample mean, σ is the standard deviation:

$$\mu = \frac{1}{m}\sum_{i=1}^{n} t_i, \quad \sigma^2 = \frac{1}{m}\sum_{i=1}^{n} t_i^2 - \mu \tag{1}$$

Z-Normalization allows us to focus on the structural feature of T, rather than its amplitude value. It addresses the problem of data stability. For instance, assume that the Euclidean Distance(ED) between two time series $T_{x,m}$, $T_{y,m}$ is expressed as follows:

$$ED_{x,y} = \sqrt{\sum_{i=1}^{m}(t_{x,i} - t_{y,i})^2} \tag{2}$$

Some little changes (e.g., the noise with a peak value) will cause an evident bias for the result, Z-Normalization is a way of smoothing the bias value.

Definition 6: *Normalized Euclidean Distance(N-ED)*, is expressed by the formula $\sqrt{\frac{1}{m}\sum_{i=1}^{m}(t_{x,i} - t_{y,i})^2}$

Definition 7: *Distance Profile DP_i* is a vector which stores the Normalized Euclidean Distance between a given subsequence/query $T_{i,m}$ and every subsequences $T'_{j,m}$ of a target Time Series T'. Formally, $DP^m_{i,j} = dist(T_{i,m}, T'_{j,m}), \forall j \in [0, n' - m + 1]$

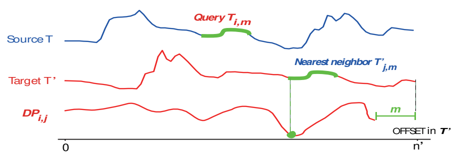

Fig. 1: Distance Profile between Query $T_{i,m}$ and target time series T', where n' is the length of T'. Obviously, $DP_{i,j}$ shares the same offset as target T'

Definition 8: *MASS*, namely Mueen's ultra-fast Algorithm for Similarity Search, computes Distance Profile based on Fast Fourier Transform(FFT), which requires just $\mathcal{O}(nlogn)$ time, other than $\mathcal{O}(nm^2)$ time in classical *N-ED* similarity search.

Definition 9: *Matrix Profile MP* is a vector of distance between subsequence $T_{i,m}$ in source T and its nearest neighbor $T'_{j,m}$ in target T'. Formally, $MP^m_i = min(DP^m_i)$, where $i \in [0, n - m + 1]$.

Like the distance profile, the matrix profile can be considered as a meta time series annotating the source time series T. The profile has a lot of interesting and exploitable properties. For example, the highest point on the profile corresponds to the time series discord, the (tied) lowest points correspond to the position of a query which has a similar matching in target time series.

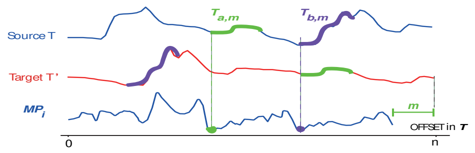

Fig. 2: Matrix Profile between Source time series T and Target time series T', where n is the length of T. Intuitively, MP_i shares the same offset as source T

B. Evaluation of candidate Shapelets

The quality of a candidate shapelet $\hat{s}$, can be assessed by its ability to separate the instances of different class in the dataset D. A prerequisite of the quality measure for $\hat{s}$, is that a set of distance $D_{\hat{s}}$ must be calculated, where $D_{\hat{s}} = D_{\hat{s},1}, D_{\hat{s},2}, ...D_{\hat{s},n}$, n is the number of T in dataset D.

Information Gain, an evaluation method based on Decision Tree, is widely adopted in previous works [3], [10], [12]. An iteration test of $D_{\hat{s}}$ is conducted to extract the best instance which brings the highest Information Gain. The distance instance will be applied as a property of candidate Shapelet, to check the inclusion between the candidate and time series. Another simple approach, is to use the F-Statistic [13], based on the difference of means in an Analysis of Variance A(NOVA). The main idea of this statistic method, is to assess the difference in distributions of $\hat{s}$ between the class distances.

C. Problem Statement

The high degree of coupling inside classical TSC algorithm [10], [12] leads to the problem of not being able to parallelize. The speed-up method such as Early Abandoning [3] is based on classical Euclidean Distance measure, which has a time complexity of $\mathcal{O}(N^2n^4)$ with several orders of magnitude higher than *MASS* [14]: $\mathcal{O}(N^2n^3logn)$. Another common trick played by previous work [3], [10], [12]: If we know $\hat{s}$ is a low-quality candidate, then any similar subsequence $\hat{s}'$ to $\hat{s}$ must also result in a low quality and therefore, a costly computation of the distance set $D_{\hat{s}'}$(evaluation of $\hat{s}'$) can be skipped. However, a candidate shapelet is evaluated by its quality ranking among all candidates of the same length. Assume that the distributed nodes have generated from dataset a collection of candidates $\hat{s}_l$, an aggregation operation between nodes is required to extract the candidate with the best quality. Extra aggregations will be made along with the iteration of candidate length. Apparently, the acceleration from the classical pruning techniques can be easily offset by the communication cost caused by the aggregation.

III. System overview

Strategies should be taken in order to make the system scalable. The main idea of our system is that the calculation should be shared and executed independently, less communication between the nodes, more powerful the algorithm would be.

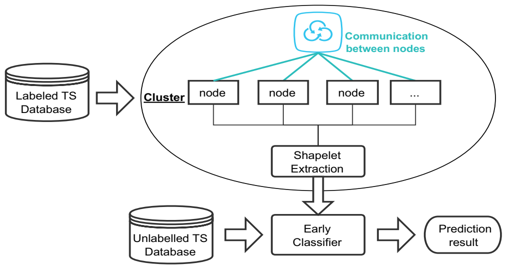

Fig. 3: System overview

A. Main structure

The conventional Time Series classification problems are tackled with nearest neighbor(1NN) algorithm due to its easy-design feature. As shown in Figure 3, on the basis of 1NN, an early classifier [15] adopted in the system allows to give the prediction result as earlier as possible without waiting for the entire sequence. The processing of labelled Time Series data requires to be flexibly arranged for nodes in the cluster. To this end, a suitable algorithm is applied here allowing assignment of computing tasks which are relatively independent of each other.

B. SMAP: Shapelet extraction on MAtrix Profile

Matrix Profile provides a meta-data which facilitates the representation of a complex correlation between two time series. *SMAP* takes the time series as the smallest processing unit between nodes, and utilizes the normalized quality to extract the most important parts in each processing unit and then merges them by an aggregation process. For this reason, the number of candidate shapelet could be greatly reduced. Moreover, a single aggregation operation is required to get the global shapelet result of different class. In *line 5*, dataset is broadcast to distributed nodes in order to reducing the communication cost caused by accessing the common data. Then, each cluster partition shares the computing takes for a set of time series. The function *computeDiscrimP* aims at computing in batches the quality of candidate shapelets. The visualized process is shown in **Figure 4**.

The batch quality of instances in a time series is defined by *Discrimination Profile*, which refers to the concept *Representative Profile*:

$$RP(T_i^C, D) = avg(MP_{T_i^C, T_j}) \tag{3}$$

where $T_j \in D^C$. Representative Profile targets thus on the minimal processing unit(i.e. time series) in SMAP, which shows a vector of Representative Power of each instance in the processing unit. To put it simply, the *Representative Power* of a subsequence in class C, is its normalized distance to the global instance cluster of class C. Intuitively, it represents the relevance between the subsequence(i.e., the candidate shapelet) and the class.

The fact that a subsequence is discriminative for its class towards others, can be expressed by the difference of Rep-

Algorithm 1: SMAP(Shapelet on MAtrix Profile)

```
Input: Dataset $D$, classSet $\hat{C}$, $k$, $offset$
Output: $\hat{S}$
1  $minLength \leftarrow 2$
2  $maxLength \leftarrow getMinLen(D)$
3  double[] $DiscmP \leftarrow []$ double[] $DistThresh \leftarrow []$
4  $\hat{S} \leftarrow \emptyset$
5  $D$.cache(); //cache all the dataset in the cluster, where each
     time series has an unique ID
6  MapPartition ($Set\ of < ID, T >: T_{set}$)
7      for $< ID, T > \in T_{set}$ do
8          for $m \leftarrow minLength\ to\ maxLength$ do
9              $DiscmP[m], DistThresh[m] \leftarrow$
                 $computeDiscmP(T, D, m, offset)$
10             $DiscmP[m] \leftarrow DiscmP[m] * \sqrt{1/l}$
11         $DiscmP \leftarrow pruning(DiscmP)$
12         emit(DiscmP, DistThresh)
13 MapAggregation ($class, (DiscmP, DistThresh)$)
14     for $c \in \hat{C}$ do
15         $\hat{S}' \leftarrow getTopk(DiscmP[c], DistThresh[c], k)$
16         for $\hat{s} \in \hat{S}$ do
17             $\hat{s}.matchingIndices \leftarrow$
                 $getMatchingIndices(\hat{s}, D)$
18         $\hat{S} \leftarrow \hat{S} \bigcup \hat{S}'$
19 return $\hat{S}$
```

resentative Power from class C to others*(OVA, one-vs-all)*. *Discrimination Profile* is then defined as follows:

$$Discm_{Profile}(T_i^C, D) = -(RP(T_i^C, D^C) - RP(T_i^C, D^{!C})) \tag{4}$$

A quality *Normalization* in *line 10* is made which allows to assess the Discrimination Power for shapelet of different length in an uniform way. Similar as the concept *Information Gain*, but Discrimination Profile is a technique more interpretable serving to assess the candidate shapelets. Moreover, in this manner, a split distance can be given directly, other than iterating every possible distance and deciding the best one with the highest Information Gain, in time $\mathcal{O}(N^2 n^2)$. A strategy to check if T contains a shapelet can be defined as the following:

$$sInT(T, \hat{s}^C) = \begin{cases} true, & if\ dist(T, \hat{s}^C) \leq RP(\hat{s}^C, D^C) \\ false, & otherwise \end{cases} \tag{5}$$

C. Acceleration strategy

The pruning function in $line 11$ is capable of eliminating the number of candidate shapelet, and then reducing the communication cost during the aggregation process. We can simply take the ”TopK” strategy, which extracts the biggest K values of *DiscrimP*. However, a such technique is far away from lightening the computation during *MapPartition* process.

Since each processing unit should be independent of each other, a tenable technique for updating the profile of a long range query could be adopted. The *Lower Bounding distance*

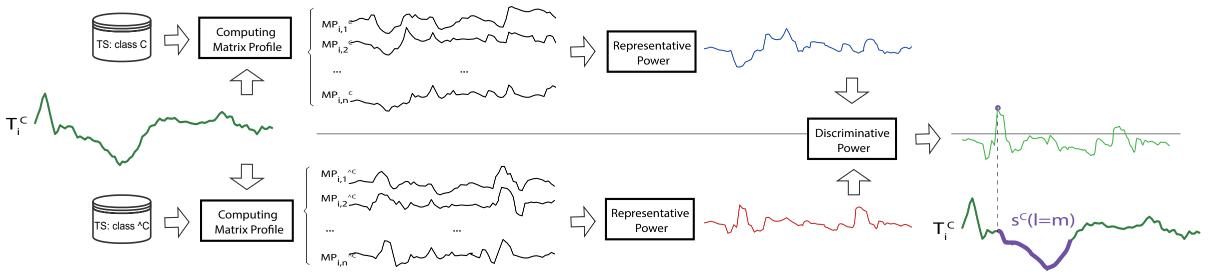

Fig. 4: Discrimination Profile Extraction

[5] is defined to estimate a minimal possible Z-Normalized Euclidean Distance between two subsequences $T_{i,l+k}$ and $T_{j,l+k}$, based on the distance already computed between $T_{i,l}$ and $T_{j,l}$. Compared to a linear time complexity of computing the exact distance, LB Distance Profile can be calculated in a constant time, which can accelerate greatly the computation of Matrix Profile in *Figure 4*. For example, from shapelet length $l = m$ to $m + 1$, the time complexity of computing the distance $d_{i,j}^{l+1}$ is $\mathcal{O}(n\frac{m(m-1)}{2}m)$, where $j \in [0, n\frac{m(m-1)}{2}]$ which represents the number of subsequences in D, n is the number of instance in D, m is the length of the longest instance in D. Accordingly, LB distance takes $\mathcal{O}(n\frac{m(m-1)}{2})$ which shows an apparent advantage when the query length is relatively long. Lower Bounding distance [5] is defined as:

$$LB(d_{i,j}^{l+k}) = \begin{cases} \sqrt{l}\frac{\sigma_{j,l}}{\sigma_{j,l+k}}, & if q_{i,j} \le 0 \\ \sqrt{l(1-q_{i,j}^2)}\frac{\sigma_{j,l}}{\sigma_{j,l+k}}, & otherwise \end{cases} \qquad (6)$$

where $q_{i,j} = \frac{\sum_{p=1}^{l}\frac{(t_{j+p-1}t_{i+p-1})}{l} - \mu_{i,l}\mu_{j,l}}{\sigma_{i,l}\sigma_{j,l}}$

Empirically, the matching subsequence $T_{j,l}$ which is the nearest neighbor of $T_{i,l}$, can deduce a longer subsequence $T_{j,l+1}$, which is probably the nearest neighbor of $T_{i,l+1}$. Assume that the matching subsequence keeps in the same position in T_{target} when query $T_{i,l}$ length increases, then the time complexity for computing the minimal distance between $T_{i,l}$ and T_{target} is $\mathcal{O}(l)$, other than $\mathcal{O}(l(n - l + 1))$. As mentioned in **Definition 9**, $MP_i^m = min(DP_i^m)$, the main idea here is to utilize LB Distance to accelerate the computation of $min(DP_i^m)$, other than computing the entire DP_i in a higher time complexity.

IV. Evaluation

The baseline of the evaluation is **USE** in [16], which utilizes the traditional method for shapelet extraction based on Information Gain. USE_{MASS} utilizes $MASS$ as Similarity Measure strategy, $SMAP_{LB}$ applies the acceleration technique based on Lower Bounding Distance. KNN(K=10) classifier is applied for all accuracy test. All implementations are based on Python3.6, with the extension PySpark of the Apache Spark. The program is tested on AWS EMR cluster (each node is with 16GB RAM, 8 vCPUs, maximal parallelism of 6, where 2 vCPUs are reserved for cluster management system). The number of nodes is adjustable.

A. *ECG medical diagnosis*

The dataset **ECG200**, from MIT-BIH Long-Term ECG Database (ltdb), is collected by two electrodes which record the brain activities in distinct body positions. Each heartbeat has an assigned label of normal or abnormal. All abnormal heartbeats are representative of a cardiac pathology known as supraventricular premature beat. ECG200 contains **100** labelled records with a fixed length of 96.

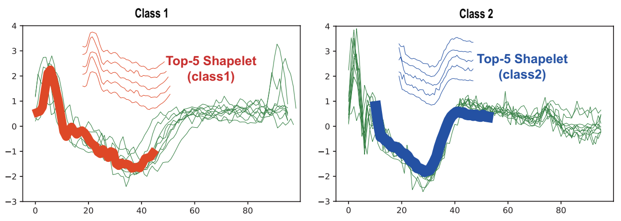

Fig. 5: Shapelets in ECG raw sequence

The shapelets extracted by SMAP and the raw time series are shown in **Figure 5**. Intuitively, shapelets of different length extracted from ECG instances, can not even be found directly by naked eyes. The performance results are shown in Table I. $SMAP_{LB}$ shows a gain in performance: **23.8X faster**, realtively higher prediction accuracy (**84%** to **76%**). We should know that the speed-up performance relies on the computing power of the cluster. We are more inclined to pay attention to its parallelization capability which can be assessed by the shuffle cost of distributed nodes. From local(1 node) to cluster mode(6 nodes), the shuffle cost increases by **106%**, the Distance Measure cost drops to **8.6%**. Obviously, considering the gain, the shuffle cost can be ignored when we expand the cluster to a larger scale.

TABLE I: Performance comparison

Algorithm	Time	Accuracy	Shuffle(L./C.)	Dist. M.(L./C.)	TopK(L./C.)
USE	2547s	76%	-	-	-
USE_{MASS}	1782s	76%	-	-	-
SMAP	139s	84%	1.52s/3.08s	1558s/135.57s	1.15s/0.27s
$SMAP_{LB}$	107s	84%	1.48s/3.19s	1198s/103.23s	1.21s/0.31s

B. *Phalanges Outlines Recognition*

To put the evaluation into larger scale context, we choose the time series dataset **PhalangesOutlinesCorrect**, which contains **1800** training records with a fixed length of 80. The target is to identify the bones of the middle finger by the outlines collected from the images. Three human evaluators labelled the output of the image outlining as correct or incorrect.

USE requires more than **12960 mins** to finish the computation task. SMAP in single node environment is more than **7 times faster** (precisely, it takes **1860 mins** in shapelets extraction). The accuracy was **86%**. **Figure 6** shows the relation between the time cost and the cluster's capacity, for both Similarity Measure and Aggregation process. The result is capable of supporting the conclusion obtained in previous experimentation.

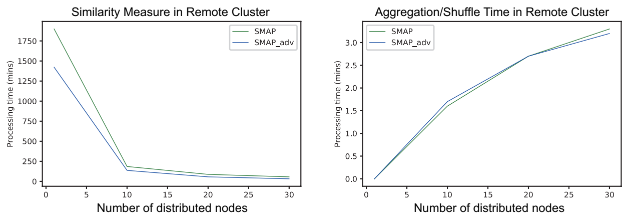

Fig. 6: Time consumed along with the node number

V. Conclusion

In this paper, we propose a novel methodology, namely SMAP, for Time Series Classification. SMAP adopts the concept Matrix Profile, and extracts the shapelet features in a scalable and interpretable manner. Within SMAP, the Discrimination Profile is defined to assess in batches the quality of candidate shapelets. On the basis of Lower Bounding, we propose also an acceleration strategy, which works appropriately for distributed environment. The satisfactory results proved the efficiency of the scalable approach, and testified its competitiveness. Different optimizations are in our planning list. Specifically, to expand SMAP for longer time series by reducing the data dimension, but meanwhile to conserve its high accuracy and scalability.

Acknowledgement

This research was supported by DATAIA convergence institute as part of the Programme d'Investissement d'Avenir , (ANR-17-CONV-0003) operated by DAVID Lab, University of Versailles Saint-Quentin, and MASTER project that has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie-Slodowska Curie grant agreement N. 777695.

References

[1] N. Ravi et al., "Activity recognition from accelerometer data," ser. IAAI'05. AAAI Press, 2005, pp. 1541–1546.
[2] L. Bao and S. S. Intille, "Activity recognition from user-annotated acceleration data." Springer, 2004, pp. 1–17.
[3] L. Ye and E. Keogh, "Time series shapelets: A New Primitive for Data Mining," *Proc. 15th ACM SIGKDD*, 2009.
[4] D.-E. Yagoubi et al., "DPiSAX: Massively Distributed Partitioned iSAX," in *Proc. ICDM 2017*, Nov. 2017, pp. 1–6.
[5] M. Linardi and Y. e. a. Zhu, "Matrix Profile X: VALMOD -Scalable Discovery of Variable-Length Motifs in Data Series," 2018.
[6] E. Keogh, K. Chakrabarti, M. Pazzani, and S. Mehrotra, "Dimensionality reduction for fast similarity search in large time series databases," *Knowledge and Information Systems*, pp. 263–286, Aug 2001.
[7] J. Lin et al., "A symbolic representation of time series, with implications for streaming algorithms," in *Proc. 8th ACM SIGMOD*, ser. DMKD '03, 2003.
[8] A. Camerra et al., "isax 2.0: Indexing and mining one billion time series," in *Proc. ICDM 2010*, 2010, pp. 58–67.
[9] T. Rakthanmanon and E. Keogh, "Fast Shapelets: A Scalable Algorithm for Discovering Time Series Shapelets," in *Proc of the 2013 SIAM International Conference on Data Mining*, 2013.
[10] A. Mueen et al., "Logical-shapelets: an expressive primitive for time series classification," *Proc. 17th SIGKDD*, 2011.
[11] K. Chang et al., "Efficient pattern-based time series classification on gpu," in *Proc. ICDM 2012*, 2012, pp. 131–140.
[12] L. Ye and E. Keogh, "Time series shapelets: a novel technique that allows accurate, interpretable and fast classification," *DMKD*, 2011.
[13] J. Lines, L. M. Davis, J. Hills, and A. Bagnall, "A Shapelet Transform for Time Series Classification," Tech. Rep., 2012.
[14] C.-C. Michael Yeh et al., "Matrix Profile I: All Pairs Similarity Joins for Time Series: A Unifying View That Includes Motifs, Discords and Shapelets," in *Proc. ICDM '16*, 2016, pp. 1317–1322.
[15] Z. Xing, J. Pei, and P. S. Yu, "Early Prediction on Time Series: a Nearest Neighbor Approach," *21st IJCAI*, pp. 1297–1302, 2009.
[16] R. Mousheimish, Y. Taher, and K. Zeitouni, "Automatic Learning of Predictive CEP Rules: Bridging the Gap between Data Mining and Complex Event Processing," in *Proc. DEBS '17*, 2017, pp. 158–169.