Triple-shapelet Networks for Time Series Classification

Qianli Ma
School of Computer Science and Engineering
South China University of Technology
Guangzhou, China
qianlima@scut.edu.cn

Wanqing Zhuang
School of Computer Science and Engineering
South China University of Technology
Guangzhou, China
scutzwq@gmail.com

Garrison W. Cottrell
Department of Computer Science and Engineering
University of California
San Diego, USA
gary@ucsd.edu

***Abstract*—Shapelets are discriminative subsequences for time series classification (TSC). Although shapelet-based methods have achieved good performance and interpretability, they still have two issues that can be improved. First, previous methods only assess a shapelet by how accurately it can classify all the samples. However, for multi-class imbalanced classification tasks, these methods will ignore the shapelets that can distinguish minority class from other classes and will tend to use the shapelets that are useful for discriminating the majority classes. Second, the shapelets are fixed after the training phase and cannot adapt to time series with deformations, which will lead to poor matches to the shapelets. In this paper, we propose a novel end-to-end shapelet learning model called Triple Shapelet Networks (TSNs) to extract multi-level feature representations. Specifically, TSN learns the most discriminative shapelets by gradient descent similar to previous methods. In addition, it learns category-specific shapelets for each class by using auxiliary binary classifiers. Finally, it uses a shapelet generator to produce sample-specific shapelets conditioned on subsequences of the input time series. The addition of category-level and sample-level shapelets to the standard model improves the performance. Experiments conducted on extensive time series data sets show that TSN is state-of-the-art compared to existing shapelet-based methods, and the visualization analysis also shows its effectiveness.**

***Index Terms*—Shapelet transformation, time series classification, temporal feature learning.**

I. Introduction

TIME series classification (TSC) is an important task in many real-world application domains and has attracted significant interest in the data mining community. In recent years, a novel approach called shapelets [1] has been applied to TSC tasks. Shapelets are discriminative subsequences of time series data. They are suitable for TSC tasks since different classes often can be distinguished by their local patterns rather than their global structure.

Shapelet-based classifiers calculate the distances between time series segments and each shapelet, and use these distances as discriminative features for classification. The original shapelet-based classifier embedded the shapelet discovery process in a decision tree and found the best shapelet at each node of the tree through enumerating all possible candidates. Although this algorithm performs quite well and is highly interpretable, it suffers from high computational complexity. An improvement to this approach called shapelet transformation (ST) managed to find the top-k shapelets in a single pass [2], and used the distances as features for standard classifiers, which both improved the classification accuracy and reduced the computational overhead.

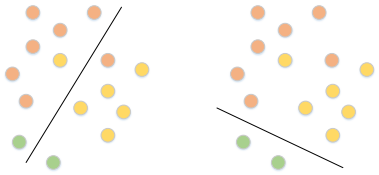

Fig. 1. The separating hyperplane on the left will obtain higher accuracy, while the one on the right will be perfect for the minority class.

Grabocka et al. [3] proposed a novel shapelet discovery approach called learning time-series shapelets (LTS) that directly learns near-optimal shapelets by optimizing the logistic regression objective function through a gradient descent algorithm. Since it learns the shapelets jointly with the classifier instead of assessing each candidate through predefined metrics and selecting the more discriminative subsequences, this approach further improves the classification accuracy.

However, for multi-class classification tasks, standard shapelet-based methods will tend to find shapelets that can separate most of the classes rather than a shapelet that can perfectly separate one class from other classes. For imbalanced datasets, the learned shapelets will tend to ignore the minority categories, because they can reduce the error the most by separating the majority classes. Shapelets that are good for separating individual categories can address the minority class problem, and hence can be very useful for imbalanced time series classification, especially when the minority category is important. To illustrate what we mean, see Fig. 1. The separating hyperplane in the left panel will be optimal for the two majority classes, while the one on the right will perfectly separate the minority class. This suggests we should learn both types of features. We can do this by using both softmax outputs to learn the dataset-level features, and binary auxiliary classifiers (one-versus-all) to learn category-specific features.

Another issue is that time series will vary in shape and minor variations in time series may lead to shapelets with poor matches to the data. Hence it would be useful to have shapelets that are specific to the data being processed. Here, it is reasonable to use a *shapelet generator* that is driven by the data itself to produce sample-specific shapelets. In this

2374-8486/19/$31.00 ©2019 IEEE
DOI 10.1109/ICDM.2019.00155

we are inspired by Dynamic Filter Networks (DFNs) [4], where convolutional filters are generated by a filter-generating network conditioned on the input. Similarly, we dynamically generate shapelets conditioned on the input time series. We train the dynamic shapelet generator by gradient descent.

Hence we propose a novel end-to-end shapelet learning model we call the Triple Shapelet Network (TSN). TSN combines the advantages of general shapelets that minimize error for major classes, category-specific shapelets that separate minority classes, and sample-level shapelets that adapt to deformations. At the dataset-level, similar to previous learning-based shapelet methods, TSN learns the most discriminative shapelets by gradient descent using a softmax output. At the category-level, TSN learns category-specific shapelets via binary categorizers for each class. This will give rise to shapelets that will act like the classifier in the right-hand panel of Fig. 1. At the sample-specific level, TSN dynamically generate shapelets conditioned on subsequences of input time series. These three different types of shapelets are then used to conduct the shapelet transformation and extract the discriminative features from the input time series. The features are input to a softmax layer to calculate the probability distribution over the categories. Since the patterns of category-level and sample-level are considered, TSN can further improve the performance of shapelet-based classifiers, especially when dealing with samples of minority classes or samples with deformations. Our contribution can be summarized as follows:

1) We propose a novel shapelet-based end-to-end model, which combines the shapelets at dataset-level, category-level and sample-level, and thus allows for more fine-grained feature extraction. 2) TSN learns category-specific shapelets to distinguish each class from the others, improving the classification accuracy of the minority classes compared to other shapelet-based methods. We use macro F1-score to measure the performance of a model on an imbalanced dataset. 3) TSN dynamically generates sample-specific shapelets conditioned on the input time series, improving the modeling flexibility by adaptively generating the shapelets. 4) The experimental results on UCR time series datasets show that the proposed model achieves state-of-the-art results when compared to other shapelet models, and the visualization analysis demonstrates the effectiveness of our model.

II. Related Work

In recent years, many efficient methods have been applied to TSC tasks, including distance-based methods, feature-based methods, ensemble-based methods and deep learning methods.

Distance-based methods measure the similarity of two given time series through well-designed similarity measures, and the classification can be done using one-nearest-neighbor (1-NN) classifiers. Examples include 1-NN with Euclidean distance, 1-NN with Dynamic Time Warping (DTW) distance [5], DD_{DTW} [6] and DTD_C [7]. Feature-based methods extract representative features from the raw time series for classification. Examples include the bag of SFA symbols (BOSS) [8], time series forest (TSF) [9], the time series bag of features (TSBF) [10] and learned pattern similarity (LPS) [11]. Shapelets [1] are also feature-based methods. The original shapelet-based classifier constructs a decision tree by recursively searching for the best shapelet at each node of the tree. Since this method suffers from high computational complexity, Lines et al. [2] proposed a method called shapelet transformation that finds the top-k shapelets in a single pass of candidates and then uses the shapelets to transform the original time series. In [12], the authors further improve the shapelet transformation method by evaluating each shapelet on how well it discriminates a single class and balances the number of shapelets per class. Grabocka et al. [3] proposed to learn shapelets directly by a gradient descent algorithm. Hou et al. [13] proposed a shapelet learning approach that learns the shapelet positions by using a generalized eigenvector method, and a fused lasso regularizer to get a sparse and "blocky" solution. This approach is significantly faster than previous shapelet-based methods. Ensemble-based methods combine different classifiers to achieve higher performance. Examples include Elastic Ensemble (EE) [14] and the collection of transformation ensembles (COTE) [15]. Although ensemble-based methods achieve better performance than their base classifiers, they suffer from high computational complexity. Deep learning methods [16] incorporate feature extraction and classification in a single end-to-end model. Examples include Multilayer Perceptrons (MLP), Fully Convolutional Networks (FCN), Residual Networks (ResNet) [17] and Encoder [18]. Although deep learning models have achieved good results, they lack interpretability. Shapelet-based methods can provide more interpretable decisions [1] since the important local patterns found from original subsequences are used to identify the category of time series.

III. Proposed Method

In this paper, we propose a novel model called the Triple Shapelet Network (TSN). Specifically, TSN learns shapelets based on discriminating all of the categories from each other (all-vs-all), discriminating each category from the others (one-vs-all) and categorizing individual time series using dynamically-generated shapelets (sample-specific shapelets). Then the learned shapelets are used to conduct the shapelet transformation on the input time series to extract the discriminative features. Finally, the extracted features are input to a softmax layer to calculate the probability distribution for each class. The general architecture of TSN is shown in Fig. 2.

A. General Shapelets

The general shapelets are use to extract the features that are useful for discriminating all of the categories from each other. Given a set of n time series $\mathbf{T} = \{\mathbf{t}_1, \mathbf{t}_2, \ldots, \mathbf{t}_n\}$, where each time series $\mathbf{t}_i$ contains m ordered real values denoted as $\mathbf{t}_i = (t_{i,1}, t_{i,2}, \ldots, t_{i,m})^T$, and a set of k general shapelets $\mathbf{S} = \{\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_k\}$. Then the shapelets can be used to transform the raw input time series to a new representation, where each attribute is the distance between the time series and one of the shapelets. The distance between the i-th time series $\mathbf{t}_i$ and the

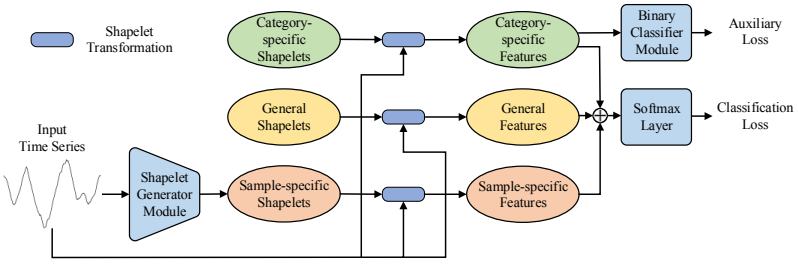

Fig. 2. General architecture of the Triple Shapelet Networks (TSNs).

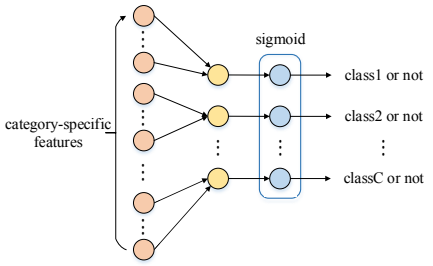

Fig. 3. An illustration of the binary auxiliary classifier.

j-th shapelet $\mathbf{s}_j$ is defined as the minimum distance between $\mathbf{s}_j$ and each subsequence of $\mathbf{t}_i$. Specifically, we denote the L-length subsequence of $\mathbf{t}_i$ starting at time step p as:

$$\mathbf{t}_{i,p:p+L-1} = (t_{i,p}, t_{i,p+1}, \cdots, t_{i,p+L-1})^T \tag{1}$$

where $\mathbf{t}_{i,p:p+L-1} \in \mathbf{R}^{L\times 1}$. Then we can calculate the distance between the i-th time series $\mathbf{t}_i$ and the j-th shapelet $\mathbf{s}_j$ by:

$$h_{i,j} = \min_{p=1,\cdots,P} \sqrt{\sum_{l=1}^{L}(t_{i,p+l-1} - s_{j,l})^2} \tag{2}$$

where $s_{j,l}$ denotes the l-th value of shapelet $\mathbf{s}_j$ and $h_{i,j}$ denotes the Euclidean distance between $\mathbf{t}_i$ and shapelet $\mathbf{s}_j$. This operation is called the shapelet transformation.

B. Category-specific Shapelets

Category-specific shapelets are used to extract features that are useful for discriminating one category from the others. In order to constrain the shapelets to learn the patterns of one specific class, we use N_c auxiliary binary classifiers, where N_c is the number of categories. The auxiliary classifiers are illustrated in Fig. 3. The units on the left are the shapelet transformation representations for each category-specific shapelet, the units in the middle are the weighted sum of these and then the sigmoid is applied. In order to simplify the hyper-parameter optimization process, if there are k shapelets used for the general shapelets, here we use $k' = \lceil k/N_c \rceil$ shapelets for each category. Specifically, using shapelets $\mathbf{S}^c = \{\mathbf{s}_1^c, \mathbf{s}_2^c, \ldots, \mathbf{s}_{k'}^c\}$ for category c, we apply the shapelet transformation to time series $\mathbf{t}_i$ by equation 2 and obtain the new representation $\mathbf{h}_i^c$. Then $\mathbf{h}_i^c$ is input to the auxiliary classifier as follows:

$$\hat{\mathbf{y}}_i^c = \mathbf{W}^c \mathbf{h}_i^c \tag{3}$$

$$P(c|\mathbf{t}_i) = sigmoid(\hat{\mathbf{y}}_i^c) \tag{4}$$

where $\mathbf{w}^c$ is the weights of the auxiliary classifier and $P(c|\mathbf{t}_i)$ denotes the probability that sample t_i belongs to category c.

C. Sample-specific Shapelets

Sample-specific shapelets are used to extract features that are adapted to the deformations in an individual sample. In order to obtain sample-specific shapelets, we use a shapelet generator to produce shapelets conditioned on the subsequences of the input time series. As shown in Fig. 4, an L-length sliding window with a stride of 1 is used to extract the subsequence of the input time series, and then one convolutional layer is used to generate the shapelets.

With sliding window length of L and stride of 1, we can obtain P ($P = m - L + 1$) subsequences of time series $\mathbf{t}_i$. And then we concatenate all the subsequences of $\mathbf{t}_i$ and denote the result as $\mathbf{O}_i$, where $\mathbf{O}_i \in \mathbf{R}^{L\times P}$ is given by

$$\mathbf{O}_i = \mathbf{t}_{i,1:L} \oplus \mathbf{t}_{i,2:L+1} \oplus \cdots \oplus \mathbf{t}_{i,m-L+1:m} \tag{5}$$

where $\oplus$ denotes the concatenation operator. Then a convolution operation is applied on $\mathbf{O}_i$ along the direction of the length with stride of 1 to obtain the generated shapelets. Let $\mathbf{s}_{i,j}^s \in \mathbf{R}^{L\times 1}$ denote the j-th generated shapelet conditioned on the i-th time series, then the shapelet $\mathbf{s}_{i,j}^s$ is defined by

$$\mathbf{s}_{i,j}^s = \mathbf{W}_j * \mathbf{O}_i + b_j \tag{6}$$

where $\mathbf{W}_j$ denotes the weights of j-th filter, b_j is the bias, $*$ denotes the convolution operation. The activation function is not used here because the value of generated shapelets should not be limited to the range of an activation function. We generate k shapelets this way, and use the shapelet transformation (Eq. 2) to extract sample-specific features.

D. Overall Loss Function

After applying the shapelet transform to the three shapelet types, we concatenate the features into a vector as follows:

$$\mathbf{h}_i^{concat} = \mathbf{h}_i \oplus \mathbf{h}_i^{category} \oplus \mathbf{h}_i^{sample} \tag{7}$$

where $\mathbf{h}_i$, $\mathbf{h}_i^{category}$ and $\mathbf{h}_i^{sample}$ are the general features, category-specific features and sample-specific features, respectively. The concatenated feature representation is fed into a softmax layer to obtain the conditional distribution over each category label as follows:

$$\hat{\mathbf{y}}_i = \mathbf{W}^{out} \mathbf{h}_i^{concat} \tag{8}$$

$$P(C|\mathbf{t}_i) = softmax(\hat{\mathbf{y}}_i) \tag{9}$$

where $\mathbf{W}_{out}$ is the weights of softmax layer, $\hat{\mathbf{y}}_i$ denotes the output vector and $P(C|\mathbf{t}_i)$ denotes the conditional label distribution of the i-th sample. Dropout is applied to $\mathbf{h}^{concat}$ to improve the generalization capability.

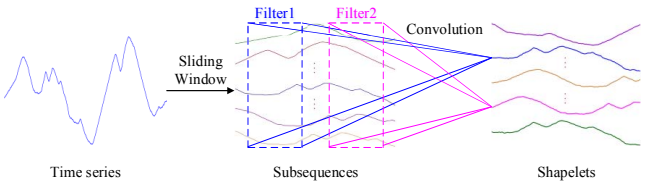

Fig. 4. The illustration of the shapelet generator.

Thus, the overall training loss L_{TSN} of TSN is defined as:

$$L_{cls} = -\sum_{i=1}^{n}\sum_{r=1}^{N_c} 1\{y_{i,r}=1\}\log\frac{\exp(\hat{y}_{i,r})}{\sum_{r'=1}^{N_c}\exp(\hat{y}_{i,r'})} \tag{10}$$

$$L_{aux} = -\sum_{i=1}^{n}\sum_{r=1}^{N_c}\left(y_{i,r}\log\frac{\exp(\hat{y}_i^r)}{1+\exp(\hat{y}_i^r)} + (1-y_{i,r})\log\frac{1}{1+\exp(\hat{y}_i^r)}\right) \tag{11}$$

$$L_{TSN} = L_{cls} + \lambda_{aux}L_{aux} \tag{12}$$

where L_{cls}, L_{aux} are classification loss and auxiliary loss, respectively. $\mathbf{y}_i$ is the target label of the i-th sample and λ_{aux} is the regularization parameter.

IV. Experiments

A. Dataset Introduction

The UCR [19] time series classification database contains 85 publicly available time series datasets which vary by the number of classes, dataset types, number of samples and time series lengths. Each data set was split into training and testing set by the provider.

B. Hyperparameter Setting

The kernel size of the convolutional layer in the shapelet generator is fixed to $3 \times P$ (P is the number of subsequences). The regularization parameter λ_{aux} is fixed to 1. The hyper-parameters of TSN are tuned through a grid search approach based on cross validation. The number of shapelets for each level is chosen from $k \in \{30, 60, 90, 120\}$. The dropout rate applied to the softmax layer is searched over $\{0, 0.25, 0.5\}$. We used two different length shapelets in each model, chosen as fractions of the time series length. We searched over the set $\{(0.1, 0.2), (0.2, 0.3), \cdots, (0.7, 0.8)\}$. We use the ADAM optimizer with an initial learning rate of 0.001.

C. Comparison to Shapelet-based Methods

TSN is compared with 9 shapelet-based methods, including the original shapelet classifier with three different shapelet quality evaluation: information gain (IG) [1], Kruskal-Wallis statistic (KW) or F-statistic (FS) [2], the fast shapelet algorithm (FSH) [20], the scalable discovery algorithm (SD) [21], the learning time-series shapelets (LTS) [3], the ultra-fast shapelet algorithm (UFS) [22], shapelet transformation with linear SVM classifier (IGSVM) [2] and the fused lasso generalized eigenvector method (FLAG) [13]. The results of these methods are collected from [13]. Following the protocol used in previous shapelet-based papers [3], [13], [20], we only report the results on 18 UCR datasets in this section.

As shown in Table I, TSN achieve the best result on 13 of the 18 datasets and the highest average rank of 2.083. Among the other shapelet-based classifiers, LTS performs well by learning the shapelets jointly with the classifiers. Due to the learning of category-specific and sample-specific shapelets, TSN outperforms LTS. To further analyze the performance, we make a pairwise comparison for each shapelet-based method against TSN. Specifically, we conducted the Wilcoxon signed rank test [23] to measure the significance of the difference. As shown in Table I, TSN is significantly better than all the shapelet-based methods at the $p < 0.05$ level except LTS, although TSN is numerically superior.

D. Comparison to State-of-the-art Methods

In this section, we compared TSN to 12 state-of-the-art TSC methods on all 85 UCR datasets. These 12 classifiers can be divided into 4 categories: 1) Distance-based methods include DD_{DTW} [6] and DTD_C [7]. 2) Feature-based methods include BOSS [8], TSF [9], TSBF [10] and LPS [11]. 3) Ensemble-based methods include EE [14] and COTE [15]. 4) Deep learning methods include MLP, FCN, ResNet [17] and Encoder [18]. The results of the deep learning methods are taken from [16], and the results of the other methods are from [24]. Due to the space limitation, we only report the summary of the results in Table II, and the full results are published on a public repository[1].

As shown in Table II, TSN achieves the best results on 24 of the 85 datasets, which is the most among all the methods. TSN achieves the average rank of 5.265, which is higher than all the comparison methods except COTE and ResNet. COTE achieves the highest average rank of 3.447. However, it is noteworthy that COTE is an ensemble model consisting of 35 different base classifiers including shapelet-based methods. Although ensemble-based methods achieve a higher accuracy than most single algorithm methods, they inevitably suffer from high computational complexity. Although the ResNet model achieves higher average rank, it is a deep learning method containing 9 convolutional layers and a large number of parameters. In contrast, TSN only uses one convolutional layer, two linear layers and has fewer parameters than ResNet. More importantly, as a shapelet-based method, TSN has better interpretability compared to the deep learning models [1].

E. Component Evaluation

To examine the strength of the three components of TSN, we evaluate each component individually against the full model, which we designate as general, category, and sample. The number and the length of shapelets used in each model are the same as those used in the full TSN model. As shown in Table III, as expected, the full TSN model obtains the highest accuracy in 16 out of the 18 datasets. The sample-specific shapelet model is better than the other two component models.

[1] https://github.com/qianlima-lab/TSN

TABLE I
ACCURACY OF TSN AND 9 SHAPELET-BASED METHODS.

Dataset	IG	KW	FS	FSH	SD	IGSVM	LTS	UFS	FLAG	TSN
Adiac	0.299	0.266	0.156	0.573	0.522	0.235	0.519	0.698	0.752	**0.798**
Beef	0.500	0.333	0.567	0.500	0.500	0.900	0.767	0.667	0.833	**0.900**
Chlorine	0.588	0.520	0.535	0.588	0.596	0.571	0.730	0.738	0.760	**0.868**
Coffee	0.964	0.857	**1.000**	0.929	0.964	**1.000**	**1.000**	0.964	**1.000**	**1.000**
Diatom	0.765	0.621	0.765	0.873	0.866	0.931	0.942	0.958	0.964	**0.990**
ECGFiveDays	0.775	0.872	0.990	0.998	0.915	0.990	**1.000**	**1.000**	0.920	**1.000**
FaceFour	0.840	0.443	0.750	0.920	0.830	**0.977**	0.943	0.932	0.909	0.955
GunPoint	0.893	0.940	0.953	0.940	0.931	**1.000**	0.996	0.987	0.967	0.967
Herring/Otoliths	0.672	0.609	0.578	0.609	0.641	0.641	0.594	0.578	0.641	**0.688**
ItalyPower	0.892	0.910	0.931	0.910	0.880	0.937	0.958	0.940	0.946	**0.969**
Lightning7	0.493	0.480	0.411	0.652	0.652	0.630	**0.790**	0.685	0.767	0.767
MedicalImages	0.488	0.471	0.508	0.647	0.660	0.552	0.713	0.711	0.714	**0.722**
MoteStrain	0.825	0.845	0.840	0.838	0.783	0.887	0.900	0.872	0.888	**0.908**
SonyAIBOSurf1	0.857	0.727	0.953	0.686	0.850	0.927	0.910	0.790	**0.929**	0.857
Symbols	0.784	0.557	0.801	0.924	0.865	0.846	0.945	0.888	0.875	**0.949**
SyntheticControl	0.943	0.900	0.957	0.947	0.983	0.873	0.973	**0.997**	**0.997**	**0.997**
Trace	0.980	0.940	**1.000**	**1.000**	0.960	0.980	**1.000**	0.960	0.990	**1.000**
TwoLeadECG	0.851	0.764	0.970	0.925	0.867	**1.000**	**1.000**	0.836	0.990	0.972
best	0	0	2	1	0	4	5	2	3	**13**
AVG rank	7.667	8.972	6.667	6.222	6.889	4.944	3.278	4.889	3.389	**2.083**
p-value	2.92E-04	1.96E-04	1.61E-03	2.93E-04	1.96E-04	2.99E-02	1.40E-01	6.42E-04	1.57E-02	-

TABLE II
ACCURACY OF TSN AND 12 STATE-OF-THE-ART METHODS.

Dataset	DD_{DTW}	DTD_C	BOSS	TSF	TSBF	LPS	EE	COTE	MLP	FCN	ResNet	Encoder	TSN
best	5	3	15	5	5	5	8	19	0	17	15	1	**24**
AVG rank	8.676	8.800	6.082	7.259	7.329	7.635	6.835	**3.529**	10.312	5.641	4.465	8.994	5.441

TABLE III
ACCURACY OF TSN AND ITS THREE COMPONENT MODELS.

Dataset	general	category	sample	TSN
Adiac	0.568	0.629	0.795	**0.798**
Beef	0.500	0.500	0.800	**0.900**
Chlorine	0.732	0.731	**0.882**	0.868
Coffee	0.893	0.893	0.964	**1.000**
Diatom	0.931	0.967	0.971	**0.990**
ECGFiveDays	0.956	0.963	**1.000**	**1.000**
FaceFour	0.943	0.886	0.920	**0.955**
GunPoint	0.873	0.967	**0.973**	0.967
Herring/Otoliths	**0.688**	0.594	0.656	**0.688**
ItalyPower	0.959	0.950	0.966	**0.969**
Lightning7	0.589	0.616	0.753	**0.767**
MedicalImages	0.630	0.584	0.720	**0.722**
MoteStrain	0.772	0.795	0.849	**0.908**
SonyAIBOSurf1	0.609	0.651	0.804	**0.857**
Symbols	0.815	0.773	0.938	**0.949**
SyntheticControl	0.977	0.743	0.993	**0.997**
Trace	0.960	0.930	0.990	**1.000**
TwoLeadECG	0.891	0.801	0.946	**0.972**

TABLE IV
THE NUMBER OF SAMPLES IN EACH CATEGORY FOR *DistPhxTW* DATASET.

category	1	2	3	4	5	6
#samples	18	19	39	13	8	42

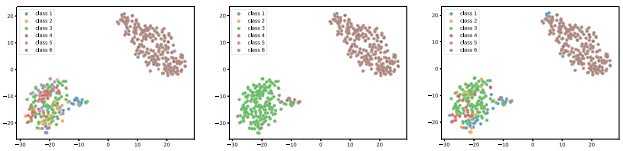

Fig. 5. The visualizations with t-SNE on (a)*DistPhxAgeGp* dataset and (b)*DistPhxTW* dataset. The labels of each subfigure from left to right are the true labels, the predicted labels of general shapelet network and the predicted labels of category-specific shapelet network.

F. Visualization Analysis

In this section, we conduct a visualization analysis to explore the effectiveness of the components of the TSN model. We first compare the general shapelet network and the category-specific shapelet network to demonstrate the effectiveness of category-specific shapelets. Then we compare the general shapelet network and the sample-specific shapelet network to show the effectiveness of sample-specific shapelets.

1) Category-specific Shapelets: For imbalanced classification tasks, the general shapelet network tends to find shapelets that are useful for discriminating the samples of the majority classes, and does not learn the shapelets that can distinguish the samples of the minority class. In contrast, the category-specific shapelet network can learn shapelets for each category, including the minority classes. To illustrate this, we employ t-SNE [25] to map the original time series into a 2-D space, and then visualize the predicted labels of both models.

We conduct experiments on the *DistPhxTW* dataset. The number of samples in each category for these dataset are shown in Table IV. As shown in Fig. 5, the general shapelet network predicts all time series as category 3 or 6, ignoring the four minority classes. In contrast, the category-specific shapelet network can identify some samples of the minority classes. To verify this result numerically, we compare the macro F1 score of these two models in Table V. The category-specific shapelet network achieves higher F1 score than the general shapelet network, proving that category-specific shapelets are more suitable for imbalanced classification.

TABLE V
THE MACRO F1 SCORE OF GENERAL SHAPELET NETWORK AND CATEGORY-SPECIFIC SHAPELET NETWORK.

Dataset	general	category-specific
DistPhxTW	0.2662	**0.3984**

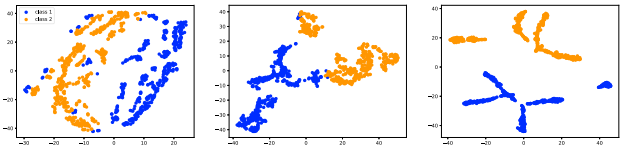

Fig. 6. The visualizations with t-SNE on *ECGFiveDays* dataset. The subfigure from left to right are the original data of the test set, the shapelet transformation of the general shapelet network and of the sample-specific shapelet network. The colors of the points indicate the true labels.

2) Sample-specific Shapelets: For the general shapelet network, the learned shapelets are fixed after training. They cannot adapt to patterns with deformations and will fail to recognize the important local patterns at test time. In contrast, the sample-specific shapelet network can produce different shapelets by the shapelet generator according to different input time series, improving the modeling flexibility. Therefore, the important local patterns of different samples from the same category can be more easily identified. To illustrate this, we employ t-SNE to map the original time series and the shapelet transformation representations into a 2-D space.

We conduct experiments on *ECGFiveDays* datasets. As shown in Fig. 6, the shapelet transformation of the general shapelet network mixes some samples that belong to different categories, resulting in the representations of category 1 and 2 being difficult to separate perfectly. In contrast, the shapelet transformation of the sample-specific shapelet network are easily separated. This is evidence that the sample-specific shapelets improve the modeling flexibility and performance.

V. CONCLUSION

In this paper, we propose a novel model called the Triple Shapelet Network (TSN) that combines general shapelets, category-specific shapelets and sample-specific shapelets to extract feature representations based on different criteria, improving the temporal feature learning ability of shapelet-based methods. The features extracted by these three kinds of shapelets are combined to generate better classification performance. Extensive experimental results show that TSN is state-of-the-art compared to existing shapelet-based methods, and the visualization analysis demonstrates its effectiveness.

ACKNOWLEDGMENT

This work is partially funded by the National Natural Science Foundation of China (Grant No. 61502174, 61872148), the Natural Science Foundation of Guangdong Province (Grant No. 2017A030313355, 2017A030313358), the Guangzhou Science and Technology Planning Project (Grant No. 201704030051, 201902010020), the Key R&D Program of Guangdong Province (No. 2018B010107002).

REFERENCES

[1] 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*. ACM, 2009, pp. 947–956.
[2] J. Hills, J. Lines, E. Baranauskas, J. Mapp, and A. Bagnall, "Classification of time series by shapelet transformation," *Data Mining and Knowledge Discovery*, vol. 28, no. 4, pp. 851–881, 2014.
[3] J. Grabocka, N. Schilling, M. Wistuba, and L. Schmidt-Thieme, "Learning time-series shapelets," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2014, pp. 392–401.
[4] X. Jia, B. De Brabandere, T. Tuytelaars, and L. V. Gool, "Dynamic filter networks," in *Advances in Neural Information Processing Systems*, 2016, pp. 667–675.
[5] D. J. Berndt and J. Clifford, "Using dynamic time warping to find patterns in time series." in *KDD workshop*, vol. 10, no. 16, 1994, pp. 359–370.
[6] T. Górecki and M. Łuczak, "Using derivatives in time series classification," *Data Mining and Knowledge Discovery*, vol. 26, no. 2, pp. 310–331, 2013.
[7] ——, "Non-isometric transforms in time series classification using DTW," *Knowledge-Based Systems*, vol. 61, pp. 98–108, 2014.
[8] P. Schäfer, "The boss is concerned with time series classification in the presence of noise," *Data Mining and Knowledge Discovery*, vol. 29, no. 6, pp. 1505–1530, 2015.
[9] H. Deng, G. Runger, E. Tuv, and M. Vladimir, "A time series forest for classification and feature extraction," *Information Sciences*, vol. 239, pp. 142–153, 2013.
[10] M. G. Baydogan, G. Runger, and E. Tuv, "A bag-of-features framework to classify time series," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 11, pp. 2796–2802, 2013.
[11] M. G. Baydogan and G. Runger, "Time series representation and similarity based on local autopatterns," *Data Mining and Knowledge Discovery*, vol. 30, no. 2, pp. 476–509, 2016.
[12] A. Bostrom and A. Bagnall, "Binary shapelet transform for multiclass time series classification," in *International Conference on Big Data Analytics and Knowledge Discovery*. Springer, 2015, pp. 257–269.
[13] L. Hou, J. T. Kwok, and J. M. Zurada, "Efficient learning of timeseries shapelets," in *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.
[14] J. Lines and A. Bagnall, "Time series classification with ensembles of elastic distance measures," *Data Mining and Knowledge Discovery*, vol. 29, no. 3, pp. 565–592, 2015.
[15] A. Bagnall, J. Lines, J. Hills, and A. Bostrom, "Time-series classification with COTE: The collective of transformation-based ensembles," *IEEE Transactions on Knowledge and Data Engineering*, vol. 27, no. 9, pp. 2522–2535, 2015.
[16] H. Ismail Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.-A. Muller, "Deep learning for time series classification: A review," *Data Mining and Knowledge Discovery*, 2019.
[17] Z. Wang, W. Yan, and T. Oates, "Time series classification from scratch with deep neural networks: A strong baseline," in *2017 International Joint Conference on Neural Networks (IJCNN)*, 2017, pp. 1578–1585.
[18] J. Serrà, S. Pascual, and A. Karatzoglou, "Towards a universal neural network encoder for time series." in *CCIA*, 2018, pp. 120–129.
[19] Y. Chen, E. Keogh, B. Hu, N. Begum, A. Bagnall, A. Mueen, and G. Batista, "The UCR time series classification archive," July 2015, www.cs.ucr.edu/~eamonn/time_series_data/.
[20] T. Rakthanmanon and E. Keogh, "Fast shapelets: A scalable algorithm for discovering time series shapelets," in *Proceedings of the 2013 SIAM International Conference on Data Mining*. SIAM, 2013, pp. 668–676.
[21] J. Grabocka, M. Wistuba, and L. Schmidt-Thieme, "Scalable discovery of time-series shapelets," *arXiv preprint arXiv:1503.03238*, 2015.
[22] M. Wistuba, J. Grabocka, and L. Schmidt-Thieme, "Ultra-fast shapelets for time series classification," *arXiv preprint arXiv:1503.05018*, 2015.
[23] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *Journal of Machine Learning Research*, vol. 7, no. Jan, pp. 1–30, 2006.
[24] A. Bagnall, J. Lines, A. Bostrom, J. Large, and E. Keogh, "The great time series classification bake off: A review and experimental evaluation of recent algorithmic advances," *Data Mining and Knowledge Discovery*, vol. 31, no. 3, pp. 606–660, 2017.
[25] L. v. d. Maaten and G. Hinton, "Visualizing data using t-SNE," *Journal of Machine Learning Research*, vol. 9, no. Nov, pp. 2579–2605, 2008.