TapNet: Multivariate Time Series Classification with Attentional Prototypical Network

Xuchao Zhang*,[1] Yifeng Gao*,[2] Jessica Lin,[2] Chang-Tien Lu[1]
[1]Discovery Analytics Center, Virginia Tech, Falls Church, VA
[2]Department of Computer Science, George Mason University, Fairfax, VA
[1]{xuczhang, ctlu}@vt.edu, [2]{ygao12,jessica}@gmu.edu

Abstract

With the advance of sensor technologies, the Multivariate Time Series classification (MTSC) problem, perhaps one of the most essential problems in the time series data mining domain, has continuously received a significant amount of attention in recent decades. Traditional time series classification approaches based on Bag-of-Patterns or Time Series Shapelet have difficulty dealing with the huge amounts of feature candidates generated in high-dimensional multivariate data but have promising performance even when the training set is small. In contrast, deep learning based methods can learn low-dimensional features efficiently but suffer from a shortage of labelled data. In this paper, we propose a novel MTSC model with an attentional prototype network to take the strengths of both traditional and deep learning based approaches. Specifically, we design a random group permutation method combined with multi-layer convolutional networks to learn the low-dimensional features from multivariate time series data. To handle the issue of limited training labels, we propose a novel attentional prototype network to train the feature representation based on their distance to class prototypes with inadequate data labels. In addition, we extend our model into its semi-supervised setting by utilizing the unlabeled data. Extensive experiments on 18 datasets in a public UEA Multivariate time series archive with eight state-of-the-art baseline methods exhibit the effectiveness of the proposed model.

1 Introduction

A time series is a set of real value observations sequentially ordered by time. A multivariate time series is a set of co-evolving time series that is typically recorded by a set of sensors simultaneously over time. With the advance of sensor technologies, The Multivariate Time Series Classification (MTSC) problem, identifying the labels for multivariate time series records, has received a great amount of attention in recent decades. Since time series data is a popular data type that exists in a wide range of research domains and applications, multivariate time series classification models have been used in many different real-world applications such as Human Activity Recognition (Minnen et al. 2006), EEG/ECG data analysis, (Bagnall et al. 2018)(Wang et al. 2015) and Motion Recognition (Rakthanmanon and Keogh 2013).

Table 1: Characteristics of Existing Models

Method	Little Domain Knowledge	Small Feature Space	Shortage of Labels	Unlabeled Data
DTW			√	√
Shapelet			√	
Bag-of-Patterns			√	
Traditional DL	√	√		
TapNet (Ours)	√	√	√	√

Most existing general time series classification approaches such as bag of patterns (Senin and Malinchik 2013) or time series shapelet (Ye and Keogh 2009) require a parsing step to convert the time series into an extensive set of subsequences or patterns as feature candidates. The large feature space generated makes the feature selection step difficult and may result in low accuracy in the multivariate case (Schäfer and Leser 2017). Recently, deep learning based methods (Karim et al. 2018)(Zheng et al. 2014) achieve promising performance in the time series classification task. These approaches can perfectly handle the issue concerning a huge feature space by learning a low-dimensional feature representation via convolutional or recurrent networks directly from raw time series data. Moreover, neural network solutions require less domain knowledge in time series data than traditional methods. But these approaches require a large amount of labeled data to train the massive model parameters. Different from the computer vision and natural language processing domains, the availability of labeled data is limited in most of the time series datasets (Bagnall et al. 2018)(Bagnall et al. 2017). For instance, the "MotorImagery" dataset (Lal et al. 2005) provided by the University of Tubingen for brain activity detection contains only 378 labeled data samples, which is arduous to train a deep learning model with thousands of model parameters (Neyshabur, Tomioka, and Srebro 2014). However, the issue of limited training labels can be handled by the traditional time series classification approach using distance-based methods such as DTW-1NN (Shokoohi-Yekta, Wang, and Keogh 2015). Table 1 presents the characteristics of existing MTSC methods. We can see that traditional time series approaches can work with limited training samples, but they usually gen-

*These two authors contributed equally.

erate a large feature space and require domain knowledge in time series data. In contrast to traditional approaches, the deep learning methods can learn low-dimensional feature representations without domain knowledge, but these approaches suffer from the limitation of training labels.

To combine the strengths of both traditional and deep learning based MTSC approaches, this paper proposes a novel multivariate Time series classification model named Time series attentional prototype network (TapNet). TapNet is capable of extracting low-dimensional features from multivariate time series with little domain knowledge and handling the shortage of labeled data. To learn the latent features from multivariate time series efficiently, we design a random group permutation method to reconstruct the dimensions of time series into groups, preceded by convolutional layers. To handle the issue of limited labeled data, we propose a novel attentional prototype network to train a low-dimensional feature representation for each time series based on their distances to the class prototype learned by a small amount of labeled samples. Since the training process is distance-based, it requires much fewer labeled samples than traditional deep neural networks. To summarize, our work has the following main contributions: 1) Propose an attentional prototype network to handle the limited training samples. 2) Learn a low-dimensional feature representation (embedding) for multivariate time series. 3) Extend our model to semi-supervised settings to utilize the unlabeled data. 4) Conduct extensive experiments on the multivariate time series datasets with a wide range of applications.

The rest of the paper is organized as follows. Section 2 discusses related work in multivariate time series classification, and Section 3 introduces the problem definition and notations. Section 4 introduces the proposed model. The experimental results on 18 UEA Archive datasets are presented in Section 5, and the paper concludes with a summary of the research in Section 6.

2 Related Work

In this section, we briefly describe recent advances in time series classification research in both multivariate and semi-supervised time series classification.

2.1 Multivariate Time Series Classification

Most work on multivariate time series classification (Baydogan and Runger 2015)(Schäfer and Leser 2017)(Wistuba, Grabocka, and Schmidt-Thieme 2015)(Baydogan and Runger 2016) follows two main directions: time series shapelet and bag-of-patterns based classification models. Wistuba et al. introduced an approach named Ultra Fast Shapelets (UFS) (Wistuba, Grabocka, and Schmidt-Thieme 2015) to efficiently select representative patterns from multivariate time series to discriminate classes. Similarly, (Karlsson, Papapetrou, and Boström 2016) introduced an approach named Generalized Random Shapelet Forests (gRSF) that generates shapelet-based decision trees via randomly selected shapelets. (Baydogan and Runger 2015) introduced an approach named Symbolic Representation for Multivariate Time series (SMTS). SMTS uses a codebook to capture the local relationships between different dimensions and uses it to classify the multivariate time series. Recently, (Schäfer and Leser 2017) introduced an approach named WEASEL-MUSE, which uses the bag of SFA (Symbolic Fourier Approximation) symbol model to classify multivariate time series. It has been shown that WEASEL-MUSE can outperform existing shapelet-based approaches (Schäfer and Leser 2017).

Recently, deep learning based methods (Karim et al. 2018)(Zheng et al. 2014) achieved promising performance in multivariate time series classification task. These models often use a LSTM layer and stacked CNN layer to extract features from the time series, and a softmax layer is then applied to predict the label. Zheng et al. introduced a model named Multi Channel Deep Convolutional Neural Network (MCDCNN) (Zheng et al. 2014), for which each univariate time series is passed through a separate CNN layer. The outputs of all univariate time series are concatenated and pass through a softmax layer to predict the label. In contrast, (Karim et al. 2018) recently proposed a model consisting of an LSTM layer and stacked CNN layer along with a Squeeze-and-Excitation block to generate latent features. Different from the conventional approaches, deep learning based methods learn the latent features by training convolutional or recurrent networks with large-scale labeled data. However, existing work does not address the problem with the (labeled) data shortage.

2.2 Semi-supervised Time Series Classification

It has been shown that in many applications, collecting labeled data is often very difficult. As a result, for time series classification, semi-supervised approaches have gained much popularity in recent years. However, most existing work on semi-supervised time series classification work focuses on univariate time series (González et al. 2018)(Wei and Keogh 2006)(Chen et al. 2013). The general approach is to use Dynamic Time Warping (DTW) (Wei and Keogh 2006)(Chen et al. 2013) to estimate labels for unlabeled time series. It has been shown that it can greatly improve the performance over the original DTW classifier. (Marussy and Buza 2013) introduced a semi-supervised approach, SUCCESS, which consists of constrained hierarchical clustering and dynamic time warping. (Begum et al. 2014) introduced a minimum description length (MDL) based stopping criterion for semi-supervised learning.

3 Problem Formulation

In this section, we begin by formulating the multivariate time series classification (MTSC) problem. Then a semi-supervised classification problem for multivariate time series (SMTSC) is formally defined.

3.1 Multivariate Time Series Classification

A multivariate time series (MTS) $X = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m\} \in \mathcal{R}^{m \times l}$ is an ordered sequence of $m \in \mathcal{N}$ streams with $\boldsymbol{x}_i = \big(x_{i,1}, \ldots, x_{i,l}\big)$, where l is the length of the time series and m is the number of multivariate dimensions. For instance, when a dust sensor collects 100 sequential particle density records in three dimensions (PM1, PM2.5, and PM10), the multivariate time series X can be represented as a matrix

with the dimension $m = 3$ and time series length $l = 100$. Each multivariate time series is associated with a class label $y \in \Omega$ from a predefined label set Ω. Given a group of multivariate time series $\mathcal{X} = \{X_1, \ldots, X_n\} \in \mathcal{R}^{n \times m \times l}$, where n is the number of time series, and the corresponding labels $\boldsymbol{y} = \{y_1, \ldots, y_n\} \in \mathcal{R}^n$ for each time series, the MTSC task is to train a classifier $f_X \mapsto y$ to predict a class label for a multivariate time series whose label is unknown.

3.2 Semi-supervised Multivariate Time Series Classification

For the semi-supervised MTSC problem, we assume the labeled samples are not sufficient to train a model for the multivariate time series classification problem. So we intend to utilize unlabeled data during the training process to help improve the overall classification performance. The training set for the semi-supervised setting is denoted as a tuple of labeled and unlabeled examples: $\big((\mathcal{X}, \boldsymbol{y}),\ \widetilde{\mathcal{X}}\big)$. The labeled portion is made up of the same input as the MTSC problem, containing tuples of time series features $\mathcal{X}$ and labels $\boldsymbol{y}$. The unlabeled portion includes a set of time series $\widetilde{\mathcal{X}} = \big\{\widetilde{X}_1, \ldots, \widetilde{X}_{\tilde{n}}\big\}$ containing only time series features without labels, where $\widetilde{X}_i$ represents the input features of the i^{th} unlabeled time series and $\tilde{n}$ is the size of unlabeled samples. The SMTSC task is to train a classifier $\tilde{f}_X \mapsto y$ to predict a class label for a multivariate time series with not only the labeled training data samples $(\mathcal{X}, \boldsymbol{y})$ but unlabeled data samples $\widetilde{\mathcal{X}}$.

4 Proposed Model

In this section, we first introduce the overall architecture of our new proposed TapNet model in Section 4.1. Then we explain the components of our model in Sections 4.2 through 4.4. Lastly, we extend our model to the semi-supervised setting (Semi-TapNet) in Section 4.5.

4.1 Model Architecture Overview

Figure 1 shows the overall architecture of our proposed model, TapNet, comprising three main components: random dimension permutation, multivariate time series encoding, and attentional prototype learning.

The input of our model is a set of multivariate time series with multiple dimensions. An example of 6-dimensional time series is shown in Figure 1. For each dimension, the time series share the same time series length. To model the interactive features between multivariate dimensions, we propose a random dimension permutation (RDP) method to randomly combine the dimensions into different groups with fixed group size. Taking the multivariate time series in Figure 1 as an example, we divide the six dimensions of the time series into three groups with different dimension permutations. A detailed description of the random dimension permutation method can be found in Section 4.2.

After the dimension permutation, a low-dimensional time series embedding is learned in the time series encoding component. Specifically, we apply both the LSTM and 1-Dimensional convolutional layers to model the sequential information of time series and the relationships between time series dimensions. After low-dimensional embeddings are learned, we use the embeddings of training samples as the input to learn the prototype for each class. Here, the class prototype is a feature representation (embedding) of each class, which contains the same embedding size as the time series. Specifically, the class prototype is a weighted combination of the training samples in the same class, where the weights of the training samples are trained by an attention layer. The intuition behind this is to learn a class prototype for each class, which has smaller distances to the data samples in the same class, but larger distances to data samples in different classes. The details of time series encoding and attentional prototype learning can be found in Sections 4.3 and 4.4, respectively.

Also, we extend the supervised TapNet model into its semi-supervised settings (Semi-TapNet). The Semi-TapNet model utilizes the feature information of unlabeled data (usually from the test set), which makes a notable improvement when the training labels are not sufficient. The details of semi-supervised attentional prototype learning can be found in Section 4.5.

4.2 Random Dimension Permutation

We propose a novel method, Random Dimension Permutation (RDP), to reorganize the time series dimensions into different groups based on random permutation orders, which helps to model the interactive features between multivariate dimensions and capture local patterns(Gao and Lin 2019)(Gao and Lin 2017).

Suppose the time series have m dimensions and they are divided into g groups, the size φ of each group can be represented as $\varphi = \lfloor \frac{m \cdot \alpha}{g} \rfloor$, where α is the scale factor parameter to control the ratio of the total dimensions in the new permutation over the original size and $\lfloor \cdot \rfloor$ represents the largest integer less than or equal to the given input. As the example shown in Figure 2, we have a time series with 6 dimensions ($m = 6$), and they are divided into $g = 3$ groups with scale factor $\alpha = 1.5$. Then the group size $\varphi = \lfloor \frac{6 \cdot 1.5}{3} \rfloor = 3$. We define σ_m as one random permutation of a set of numbers $\{1, \ldots, m\}$. To divide all the dimensions into g groups, we need to run the random permutation g times. For each permutation, we retrieve the first φ dimensions as the group candidates. To distinguish the different random permutations, we define $\sigma_m^{(i)}$ as the random permutation for the i^{th} group based on m dimensions. Then the candidates $\mathcal{G}_i$ of the i^{th} group can be defined as follows:

$$\mathcal{G}_i = \big\{\sigma_m^{(i)}(1), \ldots, \sigma_m^{(i)}(\varphi)\big\},\ \ i = 1 \ldots g. \tag{1}$$

The example in Figure 2 shows that the time series with six dimensions are divided into three groups $\big\{(4, 5, 2), (1, 2, 6), (3, 6, 4)\big\}$. Three of the six dimensions appears in two different groups, which is controlled by the scale factor parameter.

4.3 Multivariate Time Series Encoding

The time series encoding component is to learn a low-dimensional embedding $\boldsymbol{x} \in \mathcal{R}^d$ for each time series by a neural network based function $f_\Theta(X)$, where Θ is the

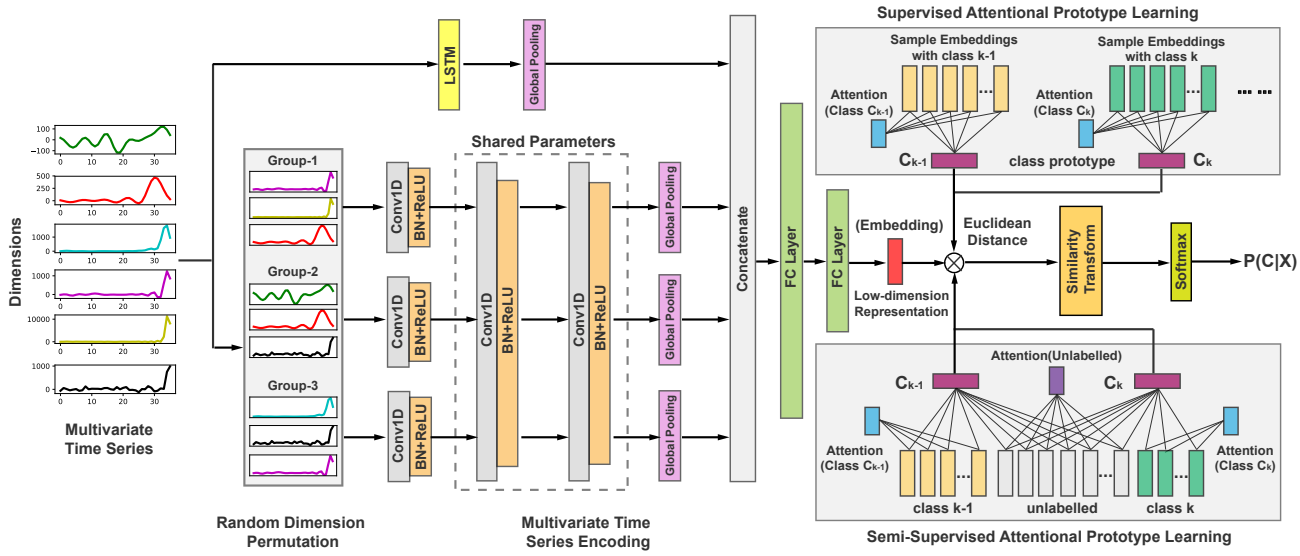

Figure 1: Overall Architecture of the TapNet Model

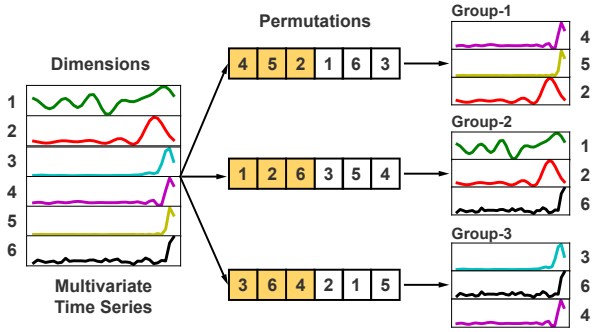

Figure 2: Example of random dimension permutation

set of function parameters and $X \in \mathcal{R}^{m \times l}$ is the time series data. To extract the sequential information as well as the multivariate features of the MTS data, we apply both a long short-term memory network (LSTM) (Sundermeyer, Schlüter, and Ney 2012) and a multi-layer convolutional network (Krizhevsky, Sutskever, and Hinton 2012). For the LSTM part, operating over the raw time series data $X \in \mathcal{R}^{m \times l}$ we obtain the contextual embedding $X_{\text{lstm}} \in \mathcal{R}^{m \times d_l}$, where d_l is the hidden dimension of LSTM with the default value 128. Then we apply a global average pooling operation on time series dimensions and get the output $X_{\text{lstm+pool}} \in \mathcal{R}^{1 \times d_l}$. For the convolutional network part, the input is the grouped permutations $\mathcal{G} = \{\mathcal{G}_1, \ldots, \mathcal{G}_g\}$. We apply three one-dimensional convolutional layers on each group $X_{\mathcal{G}_i} \in \mathcal{R}^{\varphi \times l}$. After each convolutional layer, we operate both Batch Normalization (Ioffe and Szegedy 2015) and Leaky Rectified Linear Units (Leaky ReLU) (Xu et al. 2015). Noted that we use separate parameters for the first convolutional layer but share the parameters for the second and third layers, which can help to learn separate features for each group but reduce the size of the parameters in the last two layers. We set the default value of filters for the three convolutional layers as 256, 256 and 128 and set kernels as 8, 5, and 3. For the time series with an extremely long length, we also apply a dilated convolution operation (Yu and Koltun 2015) to support exponential expansion of the receptive field without loss of long-length time series information. The output of the three convolutional layers for the i^{th} group is $X^{(i)}_{\text{conv}} \in \mathcal{R}^{d_f \times d_c}$, where d_f is the filter size of the last convolutional layer and d_c is the output dimension of convolutional operations. For each group, a global average pooling is applied on the convolutional dimensions d_c and generates the output $X^{(i)}_{\text{conv+pool}} \in \mathcal{R}^{d_f \times 1}$. Then we concatenate the results of all the groups of convolutional networks and LSTM together and get the output $X_{\text{combo}} \in \mathcal{R}^{(d_f \times g + d_l) \times 1}$. Last, we operate two fully connected layers on X_{combo} to generate the low-dimensional feature representation (embedding) of the multivariate time series $\boldsymbol{x} \in \mathcal{R}^d$, where d is the dimension of the embedding. The details of the parameter settings can be found in the Appendix.

4.4 Attentional Prototype Learning

Since the training samples in time series data can be severely limited, traditional deep learning networks may have an inductive bias (Neyshabur, Tomioka, and Srebro 2014). To address this issue, we propose a novel attentional prototype learning method by training the distance-based loss function. Specifically, in our approach, we learn a class prototype embedding (Snell, Swersky, and Zemel 2017) for each class and classify the input time series based on their distance to the prototype of each class. Let $H_k = \left[\boldsymbol{h}_1, \ldots, \boldsymbol{h}_{|S_k|}\right] \in \mathcal{R}^{|S_k| \times d}$ be a matrix of time series embeddings belonging to

the class k, where S_k represents the set of indices for data samples with class label k. Then the prototype embedding of class k can be presented by a weighted sum of individual sample embeddings as follows:

$$\boldsymbol{c}_k = \sum_i A_{k,i} \cdot H_{k,i}, \tag{2}$$

where $A_{k,i}$ is the weight of i^{th} data sample in class k and $H_{k,i}$ represents the embedding of the data sample.

Here the sample weight $A_{k,i}$ is not a predefined parameter but a trainable value according to the embeddings of time series. In particular, we regard the time series samples from the same class as a bag of independent instances. For each instance, an attention-based multi-instance pooling (Ilse, Tomczak, and Welling 2018) method is applied to learn its instance weight for the class prototype. The attention weights for the k^{th} class can be computed by the following equation:

$$A_k = \text{softmax}\left(\boldsymbol{w}_k^T \tanh\left(V_k H_k^T\right)\right), \tag{3}$$

where $\boldsymbol{w}_k \in \mathcal{R}^{u\times 1}$ and $V_k \in \mathcal{R}^{u\times d}$ are trainable parameters for the attention model and u is the size of hidden dimension for both trainable parameters. Note that we use separate parameters $\boldsymbol{w}_k$ and V_k for each class due to the assumption that the different classes may have distinct attentions on their feature spaces.

After we have the embedding vectors of class prototypes, the distribution over classes for a given time series $\boldsymbol{x} \in \mathcal{R}^d$ can be represented as a softmax over distances to the prototypes in the embedding space as follows:

$$p_\Theta\big(y = k|\boldsymbol{x}\big) = \frac{\exp\big(-D\big(f_\Theta(\boldsymbol{x}), \boldsymbol{c}_k\big)\big)}{\sum_i \exp\big(-D\big(f_\Theta(\boldsymbol{x}), \boldsymbol{c}_i\big)\big)}, \tag{4}$$

where the function $D : \mathcal{R}^d \times \mathcal{R}^d \mapsto [0, +\infty)$ is the distance function to measure the distances between two embedding vectors. The distance function can be chosen from *regular Bregman divergences* (Banerjee et al. 2005). Examples of Bregman divergences include squared Euclidean distance and Mahalanobis distance (De Maesschalck, Jouan-Rimbaud, and Massart 2000). Here we applied the squared euclidean distance function $D(\boldsymbol{z}, \boldsymbol{z}') = \|\boldsymbol{z} - \boldsymbol{z}'\|^2$ to measure the distance between the time series embeddings. The probabilities over classes are based on the similarity between class prototype and time series; therefore, we multiply -1 in front of the distance function. Then the training of our model can proceed by minimizing the negative log probability $J(\Theta) = -\log p_\Theta(y = k|\boldsymbol{x})$ of the true class via the Adam algorithm (Kingma and Ba 2014).

4.5 Semi-supervised TapNet

We now extend our supervised TapNet approach into its semi-supervised setting (Semi-TapNet) by utilizing the unlabeled data in the training phase. The unlabeled data can help to improve the estimation of the class prototype when the training data is scarce.

Let $\tilde{H} = \big[\tilde{\boldsymbol{h}}_1 \ldots, \tilde{\boldsymbol{h}}_{|\tilde{S}|}\big] \in \mathcal{R}^{|\tilde{S}|\times d}$ be a matrix of time series embeddings of unlabeled data, where $\tilde{S}$ is the set of indices for unlabeled data samples and $\tilde{\boldsymbol{h}}_i \in \mathcal{R}^{1\times d}$ is the embedding vector for the i^{th} data sample in the unlabeled set. Then the prototype embedding of class k can be presented by a weighted sum of labeled and unlabeled data as follows:

$$\boldsymbol{c}_k = \frac{\sum_i A_{k,i} H_{k,i} + \sum_i \tilde{A}_{k,i} \tilde{H}_{k,i}}{\sum_i A_{k,i} + \sum_i \tilde{A}_{k,i}} \tag{5}$$

Similar to the supervised version, we also learn instance weight for the unlabeled data samples. Here $\tilde{A}_{k,i}$ is the weight of the i^{th} unlabeled data sample for class k, and $\tilde{H}_{k,i}$ is the corresponding embedding of the data sample. The attention vector $\tilde{A}_k$ of class k for unlabeled data can be computed by the same attention mechanism as Equation (3) with different trainable parameters $\tilde{\boldsymbol{w}}_k \in \mathcal{R}^{u\times 1}$ and $\tilde{V}_k \in \mathcal{R}^{u\times d}$ as follows:

$$\tilde{A}_k = \text{softmax}\left(\tilde{\boldsymbol{w}}_k^T \tanh\left(\tilde{V}_k \tilde{H}_k^T\right)\right), \tag{6}$$

Then we apply the class prototype updated by unlabeled data into a probability distribution over class in Equation (4) to train the semi-supervised model.

5 Experiments

In this section, the performance of the proposed model, TapNet, is evaluated. We begin by introducing the evaluation setting, with details on the datasets, metrics, and baselines we use in our experiments. Then the performance of the proposed model in terms of supervised and semi-supervised classification accuracy is evaluated against several existing methods. Finally, the analyses on class prototype and random dimension permutation are elaborated. All the experiments are conducted on a single Tesla P100 GPU with 16GB memory. The source code and more experimental results are public to the research community and it can be accessed at https://github.com/xuczhang/tapnet.

5.1 Experimental Settings

Datasets We evaluate the proposed method on 18 datasets from latest multivariate time series classification archive (Bagnall et al. 2018)[1].

The archive consists of real-world multivariate time series data collected from different applications such as Human Activity Recognition, Motion classification, ECG/EEG signal classification. The dimension of tested multivariate time series ranges from two dimensions in trajectory classification data to 963 dimensions in the traffic flow classification task. The length of the time series ranges from 8 to 3,000. The datasets also have a large size range from 27 to 10,992.

Metrics For each dataset, we compute the classification accuracy as the evaluation metric. We also compute the average rank and the number of Win/Ties to compare different methods.

[1]Datasets are available at http://timeseriesclassification.com. We exclude data with extremely long length, high dimension size and in-balance split. More details can be found at https://github.com/xuczhang/tapnet.

Table 2: Performance Comparison in UEA Multivariate Time Series Dataset

Dataset	TapNet	MLSTM -FCN	WEASEL +MUSE	ED-1NN	DTW-1NN-I	DTW-1NN-D	ED-1NN (norm)	DTW-1NN-I (norm)	DTW-1NN-D (norm)
ArticularyWordRecognition	0.987	0.973	**0.99**	0.97	0.98	0.987	0.97	0.98	0.987
AtrialFibrillation	**0.333**	0.267	**0.333**	0.267	0.267	0.2	0.267	0.267	0.22
BasicMotions	**1**	0.95	**1**	0.675	**1**	0.975	0.676	**1**	0.975
CharacterTrajectories	**0.997**	0.985	0.99	0.964	0.969	0.99	0.964	0.969	0.989
FaceDetection	**0.556**	0.545	0.545	0.519	0.513	0.529	0.519	0.5	0.529
HandMovementDirection	**0.378**	0.365	0.365	0.279	0.306	0.231	0.278	0.306	0.231
Heartbeat	**0.751**	0.663	0.727	0.62	0.659	0.717	0.619	0.658	0.717
MotorImagery	**0.59**	0.51	0.5	0.51	0.39	0.5	0.51	N/A	0.5
NATOPS	**0.939**	0.889	0.87	0.86	0.85	0.883	0.85	0.85	0.883
PEMS-SF	**0.751**	0.699	N/A	0.705	0.734	0.711	0.705	0.734	0.711
PenDigits	**0.98**	0.978	0.948	0.973	0.939	0.977	0.973	0.939	0.977
Phoneme	0.175	0.11	**0.19**	0.104	0.151	0.151	0.104	0.151	0.151
SelfRegulationSCP2	**0.55**	0.472	0.46	0.483	0.533	0.539	0.483	0.533	0.539
SpokenArabicDigits	0.983	**0.99**	0.982	0.967	0.96	0.963	0.967	0.959	0.963
StandWalkJump	**0.4**	0.067	0.333	0.2	0.333	0.2	0.2	0.333	0.2
Avg. Rank	**1.15**	4.23	3.23	5.76	5.15	4.46	6.15	5.38	4.7
Wins/Ties	**12**	1	4	0	0	1	0	1	0

Comparison Methods We compare our proposed approach with eight different benchmark approaches, including the latest bag-of-patterns model based multivariate time series classification approach (Schäfer and Leser 2017), deep learning framework(Karim et al. 2018), and common distance-based classifiers. Note that it has been shown in previous work (Schäfer and Leser 2017)(Karim et al. 2018) that these approaches are equivalent or better than many other feature-based approaches such as SMTS (Baydogan and Runger 2015), gRSF (Baydogan and Runger 2016), and LPS (Karlsson, Papapetrou, and Boström 2016). The details of the benchmarks we use are provided as follows: 1) **WEASEL-MUSE** (Schäfer and Leser 2017): The latest bag-of-patterns (BOP) based framework for multivariate time series classification. We use the source code provided by the authors.[2] The approach is run under the recommended setting provided by the authors (Schäfer and Leser 2017). 2) **MLSTM-FCN** (Karim et al. 2018): The latest general deep-learning framework for multivariate time series classification. The model consists of an LSTM layer and stacked CNN layer along with a Squeeze-and-Excitation block to generate latent features. We use the source code provided by the authors.[3] The approach is run using the default parameter setting (Karim et al. 2018). 3) **1NN-ED** and **1NN-ED(norm)**: One nearest neighbor classifier with Euclidean distance with and without data normalization. This is the most popular baseline used for time series classification. 4) **1NN-DTW-i** and **1NN-DTW-i(norm)**: dimension-independent dynamic time warping with and without normalization: The nearest neighbor classifier computes distances based on the sum of Dynamic Time Warping distance for every dimension. 5) **1NN-DTW-D** and **1NN-DTW-D(norm)**: dimension-dependent dynamic time warping (Shokoohi-Yekta, Wang, and Keogh 2015) with and without normalization. This is a variation of the DTW-i approach that directly computes DTW distance based on multi-dimension points instead of treating each dimension separately.

5.2 Classification Performance Evaluation

The classification accuracy, the average rank, and number of Wins/Ties of each method is shown in Table 2. The results "N/A" in the table indicate the corresponding approach is incapable of running the results due to memory or computational issues. The best accuracy for each dataset is denoted with boldface. Overall, our approach outperforms all the other baseline approaches in terms of average rank. TapNet achieves the best overall average rank of 1.15, which is significantly better than the existing state-of-the-art approach WEASEL+MUSE with the average rank 3.23. In terms of the number of wins/ties, TapNet achieves 12 wins (or ties), the best among all eight classifiers while WEASEL+MUSE achieves 4 wins/ties and the deep learning model, MLSTM-FCN, wins (or ties) in one datasets. From the results, we found TapNet can achieve better performance in most datasets containing a small amount of data (e.g., the StandWalkJump dataset, which only contains 12 training samples). However, MLSTM-FCN often gets overall better performance in high dimensional datasets (e.g., SpokenArabicDigits). In contrast, TapNet can get better performance in both large and small datasets. Compared to WEASEL+MUSE, we found TapNet can achieve better performance in most high dimensional or large time series, whereas WEASEL+MUSE may not be able to execute due to a memory issue. This is because WEASEL+MUSE needs to generate symbols (words) for every sub-sequence per length per dimension. In the high-dimensional time series, the dictionary size can increase dramatically, which makes it hard for the approach to handle large datasets.

5.3 Semi-Supervised Performance

We next demonstrate that the Semi-TapNet model introduced in this paper can further improve the performance of classification. We evaluation the performance of our model

[2]https://github.com/patrickzib/SFA

[3]https://github.com/titu1994/MLSTM-FCN

Table 3: Performance of Semi-Supervised TapNet

Dataset (Training/Test)	TapNet	Semi-TapNet
Handwriting (150/850)	0.3565	**0.3882**
UWaveGestureLibrary (120/320)	0.894	**0.903**
ArticularyWordRecognition (275/300)	0.987	**0.993**
StandWalkJump (12/15)	**0.4**	**0.4**
JapaneseVowels (270/370)	0.965	**0.968**

on five datasets that have a significantly imbalanced training/test split.

The classification accuracy results of TapNet and Semi-TapNet in these 5 datasets are shown in Table 3. Semi-TapNet outperforms TapNet in 4 out of the 5 selected datasets. Only one dataset, StandWalkJump, shows no improvement.

Semi-TapNet can achieve better performance in datasets that have a significantly larger number of test samples compared with training samples. For example, in the Handwriting dataset, which has 850 test samples and 150 training samples, Semi-TapNet can improve classification accuracy from 0.3565 to 0.3882 (approximately 9% improvement), whereas in the ArticularyWordRecognition and JapaneseVowels datasets, which have less imbalanced training/test splits and numbers of samples, the improvement is relatively small (from 0.987 to 0.993 and from 0.965 to 0.968, respectively). The results indicate that Semi-TapNet can outperform TapNet when the dataset contains a limited amount of labeled data but a larger amount of unlabeled data.

5.4 Inspection of Class Prototype

In this section, we visualize the class prototypes and their corresponding time series embeddings to demonstrate the effectiveness of our trained low-dimensional time series embedding. We use the t-SNE algorithm (Maaten and Hinton 2008) to visualize the 300-dimension time series embedding in the form of two-dimensional images. We use different colors to isolate different classes, and we use $\circ$ and $\times$ markers to represent the training and testing samples, respectively. The class prototype is shown by the $\bigstar$ mark. Figure 3 shows the embeddings learned for the *Character-trajectories* dataset, which contains 1422 training samples and 1436 testing samples in 20 different classes. From the results, we can conclude that: 1) the distances between data samples from different classes are much larger than the distances from the same class, which means we can easily use the learned multivariate time series embeddings to classify the time series; and 2) the low-dimensional time series embeddings give us a more interpretable perspective to understand the issues of the classifier. For instance, we can see that the embeddings between class 3 and 6 are too close to separate the two classes. In fact, some testing examples in class 6 are misclassified to class 3 and vice versa. It does help to identify the issue of a classifier and take further actions such as adding more training samples in the two classes.

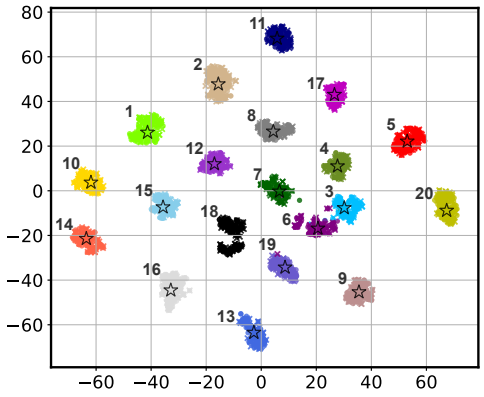

Figure 3: Class Prototype Inspection: visualize the 300-dimension multivariate time series embeddings in a two-dimensional image by t-SNE. The class prototype is marked by $\star$.

6 Conclusion

In this paper, we present a novel distance-based deep learning framework, named TapNet, for the multivariate time series classification problem. In particular, we propose a novel attentional prototype network to train the low-dimensional feature representations based on their distances to class prototype with limited training labels. Moreover, a random group permutation method is designed to learn the interactive features in multivariate dimensions combined with multi-layer convolutional networks. Additionally, we propose a semi-supervised model, Semi-TapNet, to utilize the unlabeled data in improving the classification performance when training samples are scarce. The experimental results demonstrate that our model can achieve the highest average rank on tested datasets in the public UEA archive compared to eight state-of-the-art methods.

References

Bagnall, A.; Lines, J.; Bostrom, A.; Large, J.; and Keogh, E. 2017. The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. *Data Mining and Knowledge Discovery* 31(3):606–660.

Bagnall, A.; Dau, H. A.; Lines, J.; Flynn, M.; Large, J.; Bostrom, A.; Southam, P.; and Keogh, E. 2018. The uea multivariate time series classification archive, 2018. *arXiv preprint arXiv:1811.00075*.

Banerjee, A.; Merugu, S.; Dhillon, I. S.; and Ghosh, J. 2005. Clustering with bregman divergences. *Journal of machine learning research* 6(Oct):1705–1749.

Baydogan, M. G., and Runger, G. 2015. Learning a symbolic representation for multivariate time series classification. *Data Mining and Knowledge Discovery* 29(2):400–422.

Baydogan, M. G., and Runger, G. 2016. Time series representation and similarity based on local autopatterns. *Data Mining and Knowledge Discovery* 30(2):476–509.

Begum, N.; Hu, B.; Rakthanmanon, T.; and Keogh, E. 2014. A minimum description length technique for semi-supervised time series classification. In *Integration of reusable systems*. Springer. 171–192.

Chen, Y.; Hu, B.; Keogh, E.; and Batista, G. E. 2013. Dtw-d: time series semi-supervised learning from a single example. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, 383–391. ACM.

De Maesschalck, R.; Jouan-Rimbaud, D.; and Massart, D. L. 2000. The mahalanobis distance. *Chemometrics and intelligent laboratory systems* 50(1):1–18.

Gao, Y., and Lin, J. 2017. Efficient discovery of time series motifs with large length range in million scale time series. In *2017 IEEE International Conference on Data Mining (ICDM)*, 1213–1222. IEEE.

Gao, Y., and Lin, J. 2019. Hime: discovering variable-length motifs in large-scale time series. *Knowledge and Information Systems* 61(1):513–542.

González, M.; Bergmeir, C.; Triguero, I.; Rodríguez, Y.; and Benítez, J. M. 2018. Self-labeling techniques for semi-supervised time series classification: an empirical study. *Knowledge and Information Systems* 55(2):493–528.

Ilse, M.; Tomczak, J. M.; and Welling, M. 2018. Attention-based deep multiple instance learning. *arXiv preprint arXiv:1802.04712*.

Ioffe, S., and Szegedy, C. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*.

Karim, F.; Majumdar, S.; Darabi, H.; and Harford, S. 2018. Multivariate lstm-fcns for time series classification. *arXiv preprint arXiv:1801.04503*.

Karlsson, I.; Papapetrou, P.; and Boström, H. 2016. Generalized random shapelet forests. *Data Mining and Knowledge Discovery* 30(5):1053–1085.

Kingma, D. P., and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, 1097–1105.

Lal, T. N.; Hinterberger, T.; Widman, G.; Schröder, M.; Hill, N. J.; Rosenstiel, W.; Elger, C. E.; Birbaumer, N.; and Schölkopf, B. 2005. Methods towards invasive human brain computer interfaces. In *Advances in neural information processing systems*, 737–744.

Maaten, L. v. d., and Hinton, G. 2008. Visualizing data using t-sne. *Journal of machine learning research* 9(Nov):2579–2605.

Marussy, K., and Buza, K. 2013. Success: a new approach for semi-supervised classification of time-series. In *International Conference on Artificial Intelligence and Soft Computing*, 437–447. Springer.

Minnen, D.; Starner, T.; Essa, I.; and Isbell, C. 2006. Discovering characteristic actions from on-body sensor data. In *Wearable computers, 2006 10th IEEE international symposium on*, 11–18. IEEE.

Neyshabur, B.; Tomioka, R.; and Srebro, N. 2014. In search of the real inductive bias: On the role of implicit regularization in deep learning. *arXiv preprint arXiv:1412.6614*.

Rakthanmanon, T., and Keogh, E. 2013. Fast shapelets: A scalable algorithm for discovering time series shapelets. In *proceedings of the 2013 SIAM International Conference on Data Mining*, 668–676. SIAM.

Schäfer, P., and Leser, U. 2017. Multivariate time series classification with weasel+ muse. *arXiv preprint arXiv:1711.11343*.

Senin, P., and Malinchik, S. 2013. Sax-vsm: Interpretable time series classification using sax and vector space model. In *Data Mining (ICDM), 2013 IEEE 13th International Conference on*, 1175–1180. IEEE.

Shokoohi-Yekta, M.; Wang, J.; and Keogh, E. 2015. On the non-trivial generalization of dynamic time warping to the multi-dimensional case. In *Proceedings of the 2015 SIAM international conference on data mining*, 289–297. SIAM.

Snell, J.; Swersky, K.; and Zemel, R. 2017. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*, 4077–4087.

Sundermeyer, M.; Schlüter, R.; and Ney, H. 2012. Lstm neural networks for language modeling. In *Thirteenth annual conference of the international speech communication association*.

Wang, X.; Gao, Y.; Lin, J.; Rangwala, H.; and Mittu, R. 2015. A machine learning approach to false alarm detection for critical arrhythmia alarms. In *2015 IEEE 14th international conference on machine learning and applications (ICMLA)*, 202–207. IEEE.

Wei, L., and Keogh, E. 2006. Semi-supervised time series classification. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, 748–753. ACM.

Wistuba, M.; Grabocka, J.; and Schmidt-Thieme, L. 2015. Ultra-fast shapelets for time series classification. *arXiv preprint arXiv:1503.05018*.

Xu, B.; Wang, N.; Chen, T.; and Li, M. 2015. Empirical evaluation of rectified activations in convolutional network. *arXiv preprint arXiv:1505.00853*.

Ye, L., and Keogh, E. 2009. Time series shapelets: a new primitive for data mining. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, 947–956. ACM.

Yu, F., and Koltun, V. 2015. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*.

Zheng, Y.; Liu, Q.; Chen, E.; Ge, Y.; and Zhao, J. L. 2014. Time series classification using multi-channels deep convolutional neural networks. In *International Conference on Web-Age Information Management*, 298–310. Springer.