Prototypical Inception Network with Cross Branch Attention for Time Series Classification

Jingyu Sun
NTT Software Innovation Center
NTT Corporation
Tokyo, Japan
jingyu.sun.pu@hco.ntt.co.jp

Susumu Takeuchi
NTT Software Innovation Center
NTT Corporation
Tokyo, Japan
susumu.takeuchi.sp@hco.ntt.co.j

Ikuo Yamasaki
NTT Software Innovation Center
NTT Corporation
Tokyo, Japan
ikuo.yamasaki.sm@hco.ntt.co.jp

2021 International Joint Conference on Neural Networks (IJCNN) | 978-1-6654-3900-8/21/$31.00 ©2021 IEEE | DOI: 10.1109/IJCNN52387.2021.9533440

Abstract— **Nowadays, the explosive growth of time series data and the idea of automatically classifying them brought chances of advanced analysis and machine cognitive processes in various domains. Hundreds of Time Series Classification (TSC) algorithms have been proposed during the last decade. Most of them need the large quantity of labeled training data for achieving good precision. However, we noticed that under most of the training scenarios, the large-scale supervised training datasets are not readily available. We thus proposed a few shot deep learning neural network framework PIN-BA (Prototypical Inception Network with Cross Branch Attention) for time series classification with only limited training data. We use CNN (Convolutional Neural Network) with branches of different reception windows for capturing the features of different time window scales. We design cross branch attention schemes based on prototypical networks to emphasize the crucial features' information during classification. A few experiments were conducted on the famous UCR Time Series Data Sets. Experimental results demonstrate that our proposed framework outperforms the other TSC models, such like 1NN-DTW and InceptionTime under the few shot training data scenarios.**

Keywords— Time Series Classification, Prototypical neural network, InceptionTime, Machine learning, Few shot learning

I. Introduction

With the growth of IoT and AI over the past decade, increased interest has been shown to the time series classification which can give an advanced data analysis and machine cognitive ability in various domains such like weather forecasting, financial analysis, industrial observations, health care, and so on. Meanwhile, these domains keep produce time series data of large scales in terms not only quantity but also variety which giving the previously unseen difficulty on classifying these data.

A bunch of researchers noticed that a lot of characteristics of time series could help us assign these time series a specific category. The characterization methods involve with the distribution of the values regardless of their sequential ordering, the correlation properties representing how values are correlated to themselves, and the stationarity showing how statistical properties change across a recording. Hundreds of time series classification models have been proposed including the feature-based models and models utilizing deep learning.

The feature-based models extract a bunch of features representing the time series classification patterns. Typical feature-based time series classification models include NN-DTW [1], Bag-of-Word (BoW) [2], Bag-of-Features (TSBF) [3], Bag-of-SFA-Symbols (BOSS) [4], BOSS VS (BOSS in Vector Space) [5], and HIVE-COTE [6]. We noticed that all of these models require heavy processing for the feature extraction and a large quantity of labeled data for achieving satisfying classification precision. For instance, HIVE-COTE is recognized as hugely computationally intensive and impractical to run on a real big data mining scenario for requiring training 37 classifiers while cross-validating each hyper parameters of them.

On the other hand, some other researches try to apply deep learning to time series classification. Both deep Convolutional Neural Networks (CNNs) and deep Recurrent Neural Networks (RNNs) are employed to the end-to-end classification of univariate and multivariate time series. These models include Multi-scale convolutional neural network (MCNN) [7], fully convolutional network (FCN) [8], residual network (ResNet) [9], InceptionTime [10], Echo State Networks (ESNs) [11], and ALSTM-FCN [12]. Although most of these models have proved the appropriateness of their approaches, we cannot deny that a considerable amount of labeled training data are indispensable for achieving the desired classification results. However, in most of the practical scenarios, it stays extremely difficult for people to collect these training data.

This paper analyzes these problems existing in the prior researches, and proposes a few-shot deep learning framework named PIN-BA (Prototypical Inception Network With Cross Branch Attention). PIN-BA borrows the idea of InceptionTime which utilizing CNN with multiple branches (channels) of different reception windows aiming at capturing various time scales' features. Meanwhile, we design cross branch attentions based on prototypical network framework [13] for highlighting the features of the crucial branches which promises a significant improve of both the performance and the robustness of our model. A few experiments that conducted on UCR Time Series Data Set [14] show that under the few shot training data scenarios, the performance of our model outperforms other TSC models including the traditional baseline model 1NN-DTW and the recent deep learning model InceptionTime.

II. PROBLEM DEFINITION

In information science, a time series is a series of data points indexed in time order encompassing a sequence of discrete-time data. TSC (Time Series Classification) assigns time series pattern to a specific category, for example identify a kind of heart disease based on an ECG series in health care.

The data points can be univariate or multivariate. A univariate time series $\boldsymbol{X} = [x_1, x_2, \ \dots, \ x_T]$ is an ordered set of real values. T is the number of real values. While an M-dimensional (multivariate) Time Series looks like $= [\boldsymbol{X^1}, \boldsymbol{X^2}, \ \dots, \boldsymbol{X^T}]$ where $\boldsymbol{X^i} \in \boldsymbol{R^M}$.

Time Series Classification is to constitute a dataset

$$\boldsymbol{X} = \{(X_1 \ , Y_1), (X_2 \ , Y_2), \ \dots \ (X_N \ , Y_N)\}$$

X_i **:** a univariate or multivariate TS.

Y_i: one hot label vector of K dimensions

For a dataset containing K classes, the one-hot label vector Y_i is a vector of length K where each element l_j representing the confidence of that X_i belongs to class j . Sum of all the elements in Y_i equals to 1. The classification process chooses the most confident class (highest l_j) for each X_i , and the classification result (Precision, Recall) is calculated based on these most confident classes.

III. REALATED WORK

In this section, we firstly introduce the typical models of both the feature based TSC and deep learning based TSC. And then the newest TSC base line model, InceptionTime, is illustrated in detail. At last, we point out the problem that the existing TSC models cannot perform well in few shot learning scenarios.

A. Feature Based TSC

Feature based TSC models turn out to require heavy processing or large quantity of labeled data for the feature engineering in most of the cases due to their feature extraction and comparison designs. NN-DTW, as a very strong base line, feeds a NN classifier with the Dynamic Time Warping (DTW) distances. BoW counts the representative features of time series and utilizes a classifier with the counted features as input. For handling warping at different locations and dilations, TSBF selects multiple subsequences from random locations and of random lengths, and partitions them into shorter intervals to capture the local information. For reducing the noises in time series and giving tolerance to extraneous and erroneous data, BOSS combines the extraction of substructures using the histogram created with symbolic Fourier approximation. BOSS VS proposes a vector space model to reduce time complexity of BOSS during feature calculation. Furthermore, some ensemble algorithms aiming at higher precision try to integrate different classifiers or features. Proportional Elastic Ensemble model (PROP) integrates 11 different classifiers with a weighted method. Collective of Transformation-Based Ensembles (COTE) fuses 35 classifiers together. HIVE-COTE extends COTE with a Hierarchical Vote system achieving a significant improvement over COTE. Shapelet Ensemble (SE) applies ensemble onto transformed shaplets.

B. Deep Learning Based TSC

Deep learning based TSC models give us a chance relieved from the heavy processing of feature engineering, but at the same time even more labeled training data are indispensable for the efficient training process of these models. Multi-scale convolutional neural network (MCNN), fully convolutional network (FCN), and residual network (ResNet) employs CNN for the classification. These CNNs based models were mostly inspired by the image detection models in computer vision domain. Especially, FCN shows great improvement on performance with no need to add pooling layers to reduce data dimensionality. Thus we can see unlike MCNN, FCN do not require heavy preprocessing or feature engineering, but still need a considerable amount of labeled training data. More recently, CNN models coupled with residuals such as ResNet was found being able to further improve the performance of classification [15]. The Inception Network for CNN was also applied into TSC by InceptionTime proposed in just 2019. InceptionTime is able to explore various time lengths by applying multiple convolutions with different filter lengths. Given the limitations of being hard to parallelize, RNN is still utilized by Echo State Networks (ESNs) for the time series classification task. ALSTM-FCN enhances the performance of fully convolutional networks with a nominal increase in model size and require minimal preprocessing of the data set by doing the augmentation of fully CNN with LSTM sub-modules.

C. Newest TSC Base Line Model

As a famous model in end-to-end image classification, Inception try to achieve efficient computation and reduce overfitting through a dimensionality reduction with stacked convolutions. InceptionTime, a time series classification model, is an ensemble of Inception modules, inspired by Inception-v4 architecture which performs convolution on an input with three different sizes of filters and max pooling.

Fig.1 illustrates the details of one Inception module cascaded in InceptionTime in which multiple filters are applied simultaneously to an input time series. The module includes 3 filters of different lengths allowing the network to automatically extract relevant features from both long and short time series. As shown in the figure, the bottleneck and residual connection are also applied to the network.

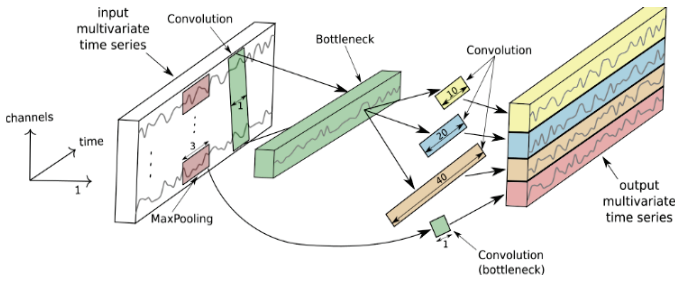

Fig. 1. Inception Module in InceptionTime

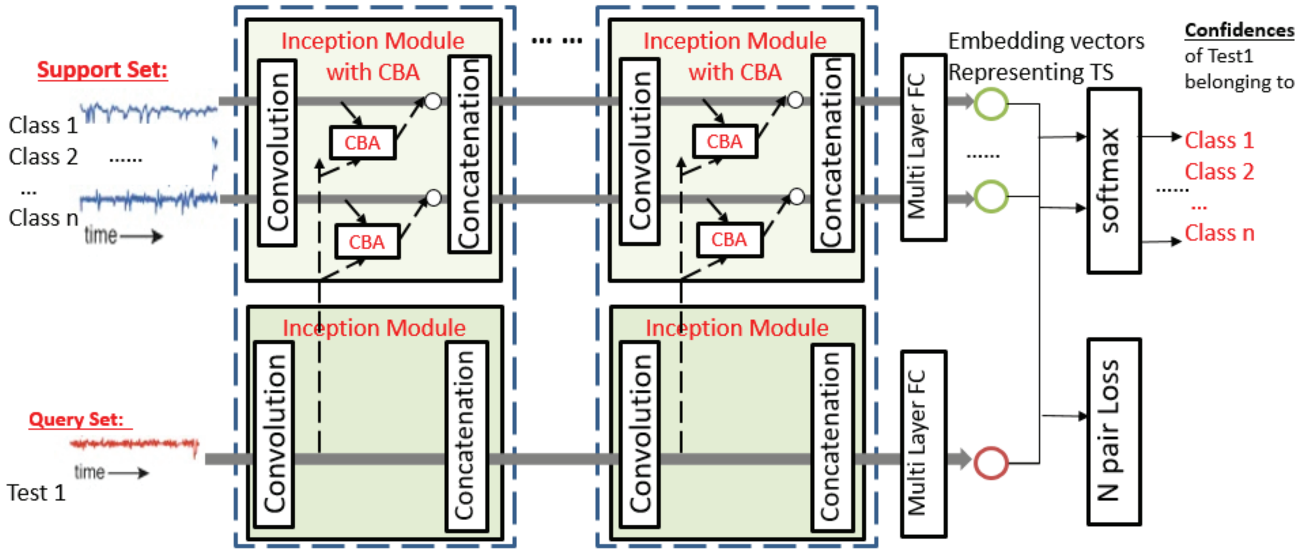

Fig. 2. PIN-BA Network

InceptionTime turns out to be highly length scalable as well as well performed on overall accuracy of other CNN based models. However, not enough experiments were conducted on the few shot training data scenarios.

D. Problem

Even though the feature based TSC models achieved good performance on some data sets, most of these models still require heavy processing for the feature extraction and a large quantity of labeled data for achieving satisfying classification precision. On the other hand, while trying to utilize deep learning for the classification task of time series, we usually need a considerable amount of training data (human labeled categories) for efficient training. However, under certain scenarios, for instance, abnormal detection, the abnormal time series which can be used for training are usually not sufficient for the normal Neural Networks' training process.

IV. PROTOTYPICAL INCEPTION NETWORK WITH CROSS BRANCH ATTENTION

Our proposed model solves the problem that the existing TSC models cannot perform well in few shot learning scenarios by introducing the architecture of Prototypical Network with specifically designed cross branch attention into the classification task of time series.

In this section, the architecture and the modules of the proposed model are illustrated in detail.

E. Proposed Model

To solve the problem of training data insufficiency, we propose a few-shot time series classification model, PIN-BA, which can automatically learn how to categorize a time series from only a few training samples. By extending the few-shot learning architecture of Prototypical Network with convolution branches of different reception windows, the model can capture the features of different time window scales under the scenario of few training data. Besides, to emphasize the information of crucial features during classification, we design cross branch attention schemes based on the branch similarities between the support set and query set of Prototypical Network.

As a prior research, Prototypical Network was originally proposed for the problems of few-shot classification, in which a classifier must generalize to new classes representation (prototype) not seen in the training set, given only a small number of examples from each new class (support data). The prototype is an M-dimensional representation $c_k \in R^M$ of each class which is calculated through an embedding function $f_\emptyset : R^D \to R^M$ with learnable parameters $\emptyset$. Each prototype is the mean vector of the support points belonging to its class:

$$c_k = \frac{1}{|S_k|} \sum_{x_i \in S_k} f_\emptyset(x_i)$$

A metric space is learned in which a test (query data) can be classified by computing distances to prototype representations of each class. Even Prototypical Network achieved good performance in few shot learning of computer vision, it has not been applied to the TSC task yet.

In our model, as shown in Fig.2, the proposed model performs an end-to-end classification task taking the time series data from support set and query set as input. For the training process, the training data x_i^j of N classes are labeled as :

$$\{(\boldsymbol{x}_1^1, c_1), \ \ldots, (\boldsymbol{x}_1^{n_1}, c_1), \ \ldots, \ \ldots, (\boldsymbol{x}_n^1, c_N), \ \ldots, (\boldsymbol{x}_n^{n_m}, c_N)\}$$

We separate the labeled data into support set and query set for every class. For N-way-K-shot training process, the support set composes of N classes, for every class, K labeled time series. In this case, n_i ; i $\in (1, \ldots, \text{m})$ all equal to K. A query set composes of Q query time series.

Firstly, for the time series from query set, we apply Inception Network consisting of multiple Inception Modules to extract the feature embedding vector of the time series. Then, we originally propose a method called CBA (Cross Branch Attention) network calculating the attention which represent the relationship of the corresponding branches between support set and query set. Finally, the calculated attentions are used for the weight recalibration of the support set's embedding in the Inception Network with CBA.

The training process is just like normal Prototypical Networks, each query time series is classified depending on its distance to the prototype of support set series. There could be many possible designs for the distance function and the classification strategy in which we choose the Euclidean distance based n-pair loss and 1-Nearest Neighbor. At the end of each training episode, the parameters of the whole network are updated by backpropagation on the loss resulted from the classification error of each query time series. In the inferring process, we apply Softmax on the calculated distances between each query set (Embedding vectors representing query TS in Fig.2) and the corresponding support set's prototype (Embedding vectors representing support TS in Fig.2) to obtain the classification results (label vectors) for each query.

F. Inception Module with CBA

The Inception Network in PIN-BA model is an ensemble of multiple Inception Modules with CBA (Cross Branch Attention) for the support set and Inception Modules without CBA for the query set. These modules have the same architecture but different initial weights. The architecture of one Inception Module with CBA are illustrated in Fig.3. We apply filters with multiple reception field sizes simultaneously to the input time series of support set and query set for extracting corresponding features from both long and short time series at the same time. Before the output of each independent parallel convolution / MaxPooling is concatenated and passed to the following layer, they are recalibrated using the CBA results from the Inception Modules of the query set.

As illustrated in Fig.3, the input of each CBA Network is the convolution result of each class from the support set and the query set. The CBA Network's output is used for the recalibration of these convolution result from support set. For the layer which is not the last layer, the recalibrated convolution results are concatenated to form the next layer's input. While for the last layer, they are concatenated and passed to a fully connect layer for forming the representation embedding vectors (prototype) of each class from the support set. All the concatenations here are performed as Depth Concat which takes inputs that have the same height and width and concatenates them along the third dimension (the channel dimension)

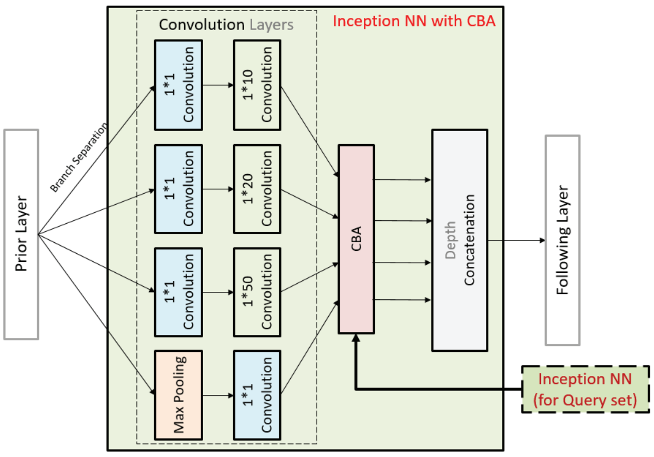

Fig. 3. Inception Module with CBA

G. Cross Branch Attention

We propose the CBA (Cross Branch Attention) to estimate the importance of different convolution branches in order to give different channel weights for recalibrating the convolution results. Fig.4 illustrated how the CBA is calculated and used for recalibrating the output of the parallel convolutions. The CBA network consists of 3 modules: F (Fully Connect), AA (Additive Attention) and S (Softmax Operation). The Additive Attention module takes the Fully Connect result of each convolution branch result from both the support set and the query set. The score of each branch i in layer j is calculated from the input convolution result $\boldsymbol{s}_i^j$ and $\boldsymbol{q}_i^j$ as following with parameter $\boldsymbol{v}_a$ and $\boldsymbol{W}_a$ to be learned during the training process.

$$Score_i^j = \boldsymbol{v}_a^T \tanh(\boldsymbol{W}_a[\boldsymbol{s}_i^j; \boldsymbol{q}_i^j])$$

Then the attention of each branch is calculated as:

$$a_i^j = \frac{\exp(Score_i^j)}{\sum_i \exp(Score_i^j)}$$

The representation vector of concept node $\boldsymbol{o}^j$ can be obtained as the depth concatenation of the weighted convolution results from each branch (i represents the jth layer in the model; i is the index of the convolution branch):

$$\boldsymbol{v}_i^j = a_i^j \boldsymbol{s}_i^j$$

$$\boldsymbol{o}^j = DepthConcat(\boldsymbol{v}_i^j)$$

Since we take the comparison result between support set and query set to calculate the attention weights of the branches, it can be expected that the branches involving with more important features for the classification process can be learned and extracted by the network during the training process.

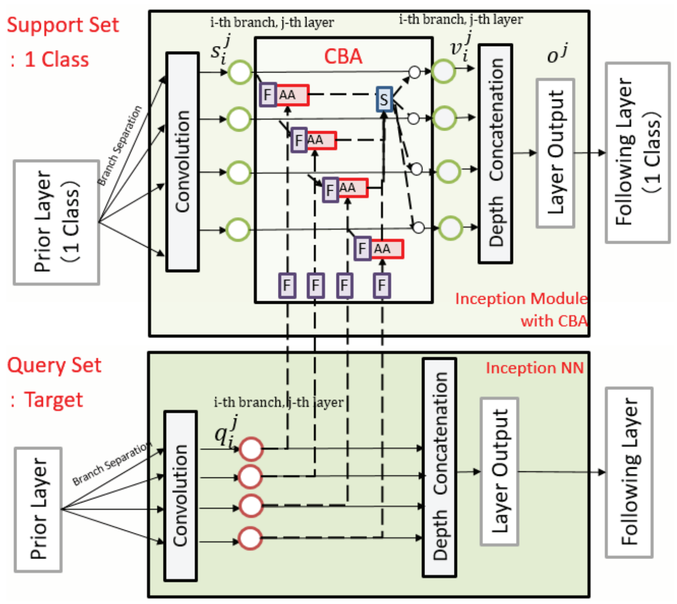

Fig. 4. Cross Branch Attention Calculation

H. Loss Function

After obtaining the vector representations of all the time series from both the support set and query set, the loss of the model can be calculated following the Multi-class N-pair Loss as below:

$$L(x,\ x^{+},\ \{x_i\}_{i=1}^{n-1},\ f) = \log(1 + \sum_{i=1}^{n-1} \exp(f^{T} f_i\ - f^{t} f^{+}))$$

x : embedding of samples in Support Set

x^{+}*: Positive Sample(Class)*

$\{x_i\}_{i=1}^{n-1}$*: Negative Samples*

n: Count of classes from support set

$f(.;\theta)$*: Embedding Kernel with parameters* θ

In the above loss function, the representation of baseline (anchor) f is compared to a positive representation f^{+} and multiple negative representations f_i for $\{x_i\}_{i=1}^{n-1}$ containing n-1 negative samples. The distance from the anchor representation to the positive representation is minimized, while the distance from the anchor and the negative representations are maximized. Obviously, this loss function put more attention on the features that can lead more dissimilarities between the vector representations of the time series.

IV. Case Study

In this section, we conducted experiments to investigate the performance enhancement of the proposed model across a range of datasets under the few training data scenario. At the same time, some other baseline models were also evaluated under the same scenario settings.

A. Data Set

For the evaluation of the proposed model, we chose 8 datasets of the UCR archive to set up for the input. The selected datasets include 8 different types of time series such as motion time series, time series generated from images and sensors, ECG time series and so on. The details of these datasets are illustrated in Table 1. We mainly chose the datasets consisting of 2-5 classes (for ECG5000, we only choose 4 classes for training since the 5th class only have 2 shot), with time series length differing from 128 to 720. The default rates of classification are around 0.5.

Table 1. Case study settings

Type	Name	Class	Length	Default rate
Simulated	CBF	3	128	0.664
Spectro	Coffee	2	286	0.464
Image	BeetleFly	2	512	0.500
Power	PowerCons	2	144	0.500
Motion	GunPoint	2	150	0.493
Image	ArrowHead	3	251	0.606
ECG	ECG5000	5	140	0.416
Device	LKA	3	720	0.667

B. Implementation Details

We train the proposed model and the baseline models (1NN-DTW, InceptionTime) on the training set introduced above with only 5 shots' training data per way (class) and report the classification error rate on the 1000 epochs. For the proposed model, we divide 5 shots into 3 shots for support set and 2 shots for query set in every training episode. For 1NN-DTW and InceptionTime, 5 shots are used directly for the training input.

In all experiments of the proposed model and InceptionTime, the parameters are set as: batch size 64, depth 6, max filter length 40, bottleneck 32, and number of filters 32. For the CBA setting in the proposed model, we let all the convolution branches to be attention wised. The output channels, which is also the final embedding dimensions of the time series' representation are set as 20. In addition, we apply dropout of 0.2 for the fully connect layer which generates the final embedding vector for every time series. Besides, all the parameters in our proposed model are optimized using Adam with a learning rate of 0.0005.

C. Results

The pairwise plot of error rate in Fig.5 compares PIN-BA to the baseline model, 1NN-DTW model. We found that PIN-BA showed an obvious improvement over 1NN-DTW. Specifically, the Win/Tie/Loss result turned out to be 6/1/1 in favor of PIN-BA against 1NN-DTW.

Table 2. Error Rate on different data sets

	PIN-BA	InceptionT-ime	InceptionT-ime	1NN-DTW	1NN-DTW	1NN-BOSS (by paper)	1NN-ED (by paper)
	(5shot)	(5shot)	(Full)	(5shot)	(Full)	(Full)	(Full)
CBF	0.009	0.070	0.020	0.050	0.003	0.000	0.150
Coffee	0.000	0.000	0.000	0.040	0.000	0.000	0.000
BeetleFly	0.190	0.250	0.200	0.400	0.300	0.100	0.250
PowerCons	0.083	0.140	0.060	0.120	0.078	-	0.070
GunPoint	0.133	0.140	0.000	0.180	0.087	0.000	0.086
ArrowHead	0.332	0.430	0.160	0.330	0.200	0.140	0.200
ECG5000	0.297	0.310	0.200	0.270	0.075	0.060	0.075
LKA	0.456	0.520	0.100	0.570	0.200	0.280	0.510

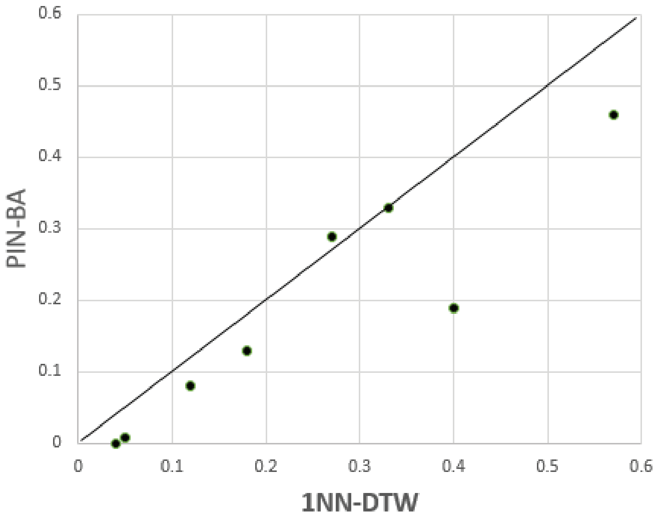

Fig. 5. Classification error rate of 1NN-DTW and PIN-BA (proposed)

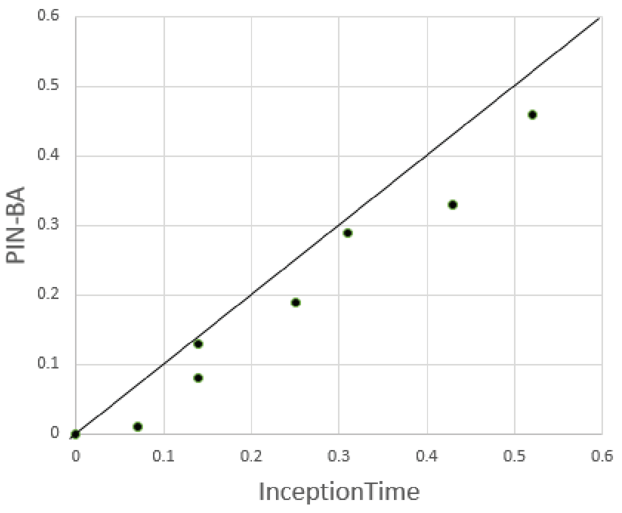

Fig. 6. Classification error rate results of InceptionTime and PIN-BA(proposed)

The biggest improvement was on the BeetleFly dataset, we achieved enhancement from 0.40 to 0.19 on error rate.

Only one dataset on which 1NN-DTW outperformed PIN-BA is ECG5000. We will investigate the reason for this result in the discussion section.

To visualize the difference between the PIN-BA and InceptionTime, which is an ensemble of 6 Inception Module without attention scheme, Fig. 6 depicts the pairwise error rate plot of PIN-BA against InceptionTime for each of the 8 UCR datasets. The results show a Win/Tie/Loss of 8/0/0 in favor of PIN-BA.

The biggest improvement was on the ArrowHead dataset, the error rate was reduced from 0.43 to 0.33.

Comparing to other dataset, we found that PIN-BA failed to achieve an obvious improvement on LKA (Large Kitchen Applicant) dataset. We will analyze the reason and possible approaches for this problem in the discussion section.

We conducted experiments not only on few shot training data but also on full dataset with InceptionTime and 1NN-DTW models for analyze the characteristics of each dataset. The results are as the Table.2. We found that under the few training data scenario, expect ECG5000 and LKA, PIN-BA achieved significant improvement compared with InceptionTime and 1NN-DTS. However, these 2 models still achieve better performance than PIN-BA if they are feed by full dataset.

D. Ablation Study

To evaluate the contribution of the proposed Cross Branch Attention Network, we perform ablation studies on the proposed model. We removed the attention network and replaced it as an average of the attributes' vectors; the results are shown in Fig.7.

We observe that attention network contribute to the performance on half of the datasets. For the CBF dataset, without the attention network, error rate increased from 0.09 to

0.11, while for the datasets BeetleFly, ECG5000 and GunPoint, the error rate also increased about 0.01.

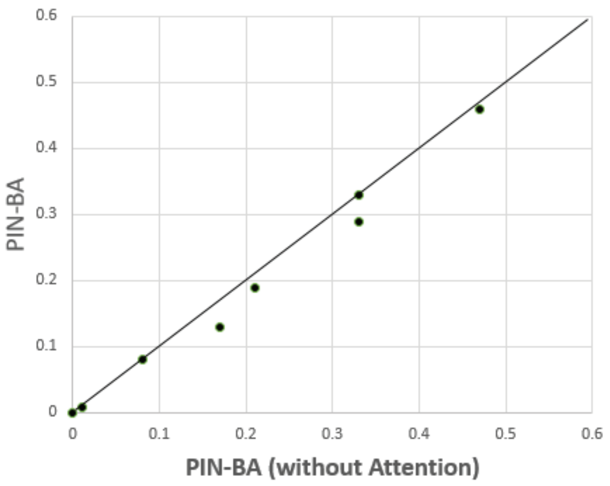

Fig. 7. Classification error rate results of PIN-BA and PIN-BA (without Attention)

V. Discussion

In this section, we discuss the possible reasons and solutions for the low performance of PIN-BA on LKA and ECG5000 datasets.

As listed up in Table.2, even with the same classification strategy, 1NN-DTW outperforms 1NN-ED dramatically with an decrease of error rate 0.51 to 0.2. And as we know, DTW performs well than ED especially for the time series of shifting phases. And since PIN-BA shows a low performance on LKA, this implies that PIN-BA may be not good at capturing the features with shifting phases. Thus we suggest having a test on full dataset with DTW and ED confirming if the problem of shifting phases is serious in the dataset or not before conducting few shot learning with PIN-BA.

From the evaluation results of dataset ECG5000, 1NN-BOSS achieved an error rate of 0.06 much lower than the error rate of 0.2 gained by InceptionTime. As we know, BOSS applies the features of Fourier transformation results from time series to the distance calculation; while InceptionTime performs an end-to-end classification directly towards the time series data. Thus we infer that Fourier features need to be considered if higher accuracy is required.

VI. Conclusion

In this paper, we propose a few-shot classification model for time series, which can automatically learn the crucial features involving with the classification task from only a few training samples per class. Specially, the proposed model applies the Prototypical Network in computer vision to time series classification and designs an attention estimation network CBA learning the weights for different branches of various reception fields.

Experimental results on few-shot training samples (5 shot) validate that our model can identify important features for the classification, and the performance of our model stays high comparing with other baseline models with zero parameter setting and few-shot training samples.

In the future, we are about to update our model considering datasets with shifting phases to confirm further feasibility and effectiveness of the proposed architecture.

References

[1] 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 31(3), pp.606–660, 2017

[2] J. Wang, P. Liu, F.H. Mary, S. Nahavandi and A. Kouzani, “Bag-of-words representation for biomedical time series classification,” Biomedical Signal Processing and Control, Volume 8, Issue 6, November 2013, Pages 634-644

[3] M. Baydogan, G. Runger and E. Tuv, “A Bag-of-Features Framework to Classify Time Series,” IEEE Transactions on Pattern Analysis and Machine Intelligence 35(11):2796-802, 2013

[4] P. Schäfer, “The boss is concerned with time series classification in the presence of noise,” Data Mining and Knowledge Discovery 29(6):1505–1530, 2015

[5] P. Schäfer, “Scalable time series classification,” Data Mining and Knowledge Discovery, 30(5):1273–1298, 2016. ISSN 1573756X

[6] J. Lines, S. Taylor, and A. Bagnall, “Time series classification with hive-cote: The hierarchical vote collective of transformation-based ensembles,” ACM Transactions on Knowledge Discovery from Data (TKDD), 12(5):52, 2018

[7] Z. Cui, W. Chen and Y. Chen, “Multi-scale convolutional neural networks for time series classification,” ArXiv 1603.06995, 2016

[8] Z. Wang, W. Yan and T. Oates, “Time series classification from scratch with deep neural networks: A strong baseline,” International Joint Conference on Neural Networks, pp 1578-1585, 2017

[9] K. He, X. Zhang, S. Ren and J. Sun, ”IEEE Conference on Computer Vision and Pattern Recognition, pp 770-778, 2016

[10] H.I. Fawaz, B. Lucas, G. Forestier, C. Pelletier, D.F. Schmidt, J. Weber, G. I. Webb, L. Idoumghar, P.A. Muller and F. Petitjean, “InceptionTime: Finding AlexNet for Time Series Classification,” Data Mining and Knowledge Discovery volume 34, pages1936–1962, 2020

[11] X. H. Yao and Z.S. Wang, “Broad Echo State Network for Multivariate Time Series Prediction,” Journal of the Franklin Institute 356(9), 2019

[12] F. Karim, S. Majumdar, H. Darabi and S. Harford, “Multivariate LSTM-FCNs for time series classification, ” Neural Networks Volume 116, Pages 237-245, 2019

[13] J. Snell, K. Swersky and R. Zemel, “Prototypical Networks for Few-shot Learning,” Advances in Neural Information Processing Systems 30 (NIPS 2017)

[14] H.A. Dau, A. Bagnall, K. Kamgar, C.C.M. Yeh, Y. Zhu, S. Gharghabi, C.A. Ratanamahatana and E. Keogh, “The ucr time series archive.” ArXiv, 2018

[15] F. H. Ismail, G. Forestier, J. Weber, L. Idoumghar and P. A. Mulle r“Deep learning for time series classification: a review,” Data Mining and Knowledge Discovery 33 (4), 917–963, 2019