


Few-shot class-incremental learning for Medical time series classification

Le Sun, Mingyang Zhang, Benyou Wang and Prayag Tiwari , Senior Member, IEEE

Abstract—Continuously analyzing medical time series as new classes emerge is meaningful for health monitoring and medical decision-making. Few-shot class-incremental learning (FSCIL) explores the classification of few-shot new classes without forgetting old classes. However, little of the existing research on FSCIL focuses on medical time series classification, which is more challenging to learn due to its large intra-class variability. In this paper, we propose a framework, the Meta self-Attention Prototype Incrementer (MAPIC) to address these problems. MAPIC contains three main modules: an embedding encoder for feature extraction, a prototype enhancement module for increasing inter-class variation, and a distance-based classifier for reducing intra-class variation. To mitigate catastrophic forgetting, MAPIC adopts a parameter protection strategy in which the parameters of the embedding encoder module are frozen at incremental stages after being trained in the base stage. The prototype enhancement module is proposed to enhance the expressiveness of prototypes by perceiving inter-class relations using a self-attention mechanism. We design a composite loss function containing the sample classification loss, the prototype non-overlapping loss, and the knowledge distillation loss, which work together to reduce intra-class variations and resist catastrophic forgetting. Experimental results on three different time series datasets show that MAPIC significantly outperforms state-of-the-art approaches by 27.99%, 18.4%, and 3.95%, respectively.

Index Terms—Health monitoring, Medical decision-making, Few-shot class-incremental learning, Medical time series classification.

I. INTRODUCTION

CLASSIFICATION of medical time series is essential for signal-based disease diagnosis. Although images are widely used in the medical field, doctors often diagnose diseases based on one-dimensional physiological signals. When an intelligent medical decision system is used to monitor

a patient's physiological signals, it continually encounters new classes that have never been seen before. These classes are brand-new; thus there is not much data available. The decision system should be able to classify new classes without forgetting the old ones. Therefore, dynamic medical time series classification is a few-shot class-incremental learning (FSCIL) problem. An ideal FSCIL system for medical time series classification is shown in Fig. 1. Achieving this objective would be extremely advantageous for the deployment of such an automated disease diagnosis and monitoring system.

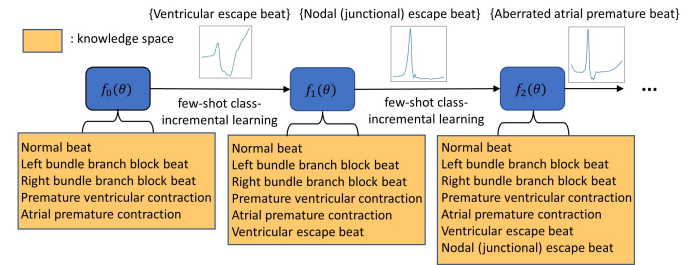


Fig. 1. Few-shot class-incremental learning for Medical time series classification. The model $f(\theta)$ learns new signal classes sequentially at incremental stages while not forgetting the old classes. All signal classes so far seen can be classified using the modified model.

A lot of approaches [1]–[3] have been proposed to address catastrophic forgetting in the computer vision area. However, there is little work solving the FSCIL problem for time series. There are three main challenges on FSCIL for time series classification:

- **Catastrophic forgetting.** From the perspective of artificial intelligence, recent researches [4]–[7] have suggested employing more sophisticated deep learning-based algorithms to improve model performance; however, these methods can only be trained on static datasets and cannot process newly accessible classes. As a result, due to the inaccessibility of old data, the classification accuracy could significantly decline, which is also known as catastrophic forgetting [8]. It is an alternative approach to retraining from scratch after observing a new class. However, it is time-consuming and requires a significant amount of computing and memory, particularly for large-scale medical time series datasets. As new classes emerge continually in FSCIL, the model suffers from forgetting old classes.
- **Lack of new class data.** During time series collection, new classes may appear. Due to the limited training data for new classes, it is challenging to guarantee the

This work is partially supported by the Priority Academic Program Development of Jiangsu Higher Education Institutions, the National Natural Science Foundation of China (Grant No. 62206231), and the Major Key Project of PCL (Grant No. PCL2022A03, PCL2021A02, PCL2021A09). (Corresponding Author: Prayag Tiwari)

Le Sun and Mingyang Zhang are with the Engineering Research Center of Digital Forensics, Ministry of Education, Nanjing University of Information Science and Technology; and the Department of Jiangsu Collaborative Innovation Center of Atmospheric Environment and Equipment Technology (CICAET), Nanjing University of Information Science & Technology, Nanjing, 210044, China (e-mail: LeSun1@nuist.edu.cn and 20211220041@nuist.edu.cn)

B. Wang is with the SRIBD & SDS, The Chinese University of Hong Kong, Shenzhen, China (email: wangbenyou@cuhk.edu.cn)

P. Tiwari is with the School of Information Technology, Halmstad University, Sweden (email: prayag.tiwari@ieee.org)

classification accuracy of all classes.

- **Intra-class variation.** Time series signals show more intra-class variations compared with image data. Take Electrocardiogram (ECG) as an example, under different temporal or physical settings, the same patient's ECG waveforms vary greatly. As a result of individual variations in ECG waveforms, identical heart disease may have different waveforms in different persons, whereas two distinct cardiac disorders may have similar waveforms [9].

To solve the above problems, we propose a framework called **Meta self-Attention Prototype InCrementer (MAPIC)**, for time series classification in FSCIL. MAPIC is composed of three parts: the embedding encoder module (EE), the prototype enhancement module (PE), and the distance-based classifier (DC). EE is used to extract features. It maps a sample to a high-dimensional embedding. PE and DC reduce the intra-class variation.

FSCIL usually includes a base stage and several incremental stages. To prevent catastrophic forgetting at incremental stages, we design a replay-based method to store part of the data of old classes for training. In addition, a parameter protection strategy is proposed to prevent the parameters crucial to old classes from updating. EE will be frozen after training in the base stage, and only the PE is updated at incremental stages. DC is non-parametric and has no need to update.

To address the lack of new class data at incremental stages, we adopt meta-learning for training MAPIC. On account of the limited data available, MAPIC uses episodes of data rather than batches of data to mimic the N-way-K-shot scenario, which has demonstrated remarkable potential in few-shot learning [10]. Each episode contains a support set and a query set. The former is used to get prototypes, and the latter is used to calculate loss functions and update the MAPIC.

PE and DC are designed to reduce intra-class variation. Intuitively, DC classifies an unseen sample by comparing it with labeled instances. The mean of all sample embeddings of a class in the support set is used to represent the prototype of the class. MAPIC makes predictions based on the Euclidean distance between query embeddings and prototypes. Training the MAPIC with a distance-based classifier can explicitly reduce intra-class variations [11]. However, due to the lack of new class data in each incremental stage, the direct average of feature embedding in the support set cannot be fully representative and is quite noise-sensitive. PE takes all the raw prototypes as the input of the self-attention mechanism, and lets each prototype perceive inter-class relations, thus enhancing the expressiveness of prototypes through information aggregation.

We design a composite loss function to train the MAPIC, that contains sample classification loss, prototype non-overlapping loss, and knowledge distillation loss. The sample classification loss is a variant of the cross-entropy loss, in which the prediction probabilities are obtained by normalizing the distances between query embeddings and prototypes. At the base stage, only the sample classification loss is used to update the MAPIC. However, as new classes emerge in incremental stages, their prototypes may overlap in metric

space with the ones of old classes. We use a nonlinear monotonic decreasing function as a prototype non-overlapping loss. By optimizing it, the smallest distance among prototypes will increase, leading to an increase in all prototype distances. The distance between prototypes becomes larger, further enhancing the classification accuracy. The knowledge distillation loss utilizes Kullback-Leibler (KL) divergence to measure the similarity of two groups of probabilities. MAPIC uses the previous stage's classification results as the teacher to guide the representation learning at the current stage, which is also beneficial for mitigating catastrophic forgetting. Three loss functions serve as a composite loss function at incremental stages.

The following is a summary of the contribution of this work:

- 1) We propose the MAPIC framework to solve the few-shot class-incremental learning of medical time series. To the best of our knowledge, we are the first to investigate FSCIL for medical time series classification.
- 2) We design a distance-based classifier and a prototype enhancement module to reduce the intra-class variations.
- 3) We design a composite loss function containing the sample classification loss, the prototype non-overlapping loss, and the knowledge distillation loss. The three losses work together to reduce intra-class variation, enhance inter-class variation and avoid catastrophic forgetting.
- 4) We conduct extensive experiments on the MIT-BIH, FaceAll, and UWave datasets, which show that our framework significantly outperforms state-of-the-art methods.

The rest of this paper is structured as follows: In Section II, the related work is introduced. Section III outlines the FSCIL problem statement. The methodology of MAPIC is described in Section IV. Comprehensive experiments and outcomes are presented in Section V. Section VI summarize all the work.

II. RELATED WORK

A. Few-Shot Learning

Few-shot learning (FSL) [11] seeks to create a model that can identify previously unseen classes based on a limited number of training examples. FSL can often be classified into two categories: metric-based methods and optimization-based methods. The core of metric-based methods [12], [13], [16] is "learning to compare". It is inspired by [17], in which if a model can distinguish the similarity of two classes, it can classify new classes it has never seen by comparing them with the identified classes. [11] proposes that without using meta-learning, the performance of a basic baseline technique using a distance-based classifier is competitive with respect to other sophisticated algorithms. Optimization-based methods [18], [19] aim to train a framework for better generalization and fast model convergence with limited exemplars.

B. Incremental Learning

Incremental learning (IL) is also called continual learning [20], or lifelong learning [21]. From [22], IL can be divided into three scenarios, class-incremental learning (CIL) [14],

TABLE I
COMPARISON WITH THE RELATED WORKS

Paper	Method	Training Data	Ability to classify new classes	Few-Shot Class-Incremental Learning
Snell et al. [12]	meta-learning + Euclidean distance classifier	Image	yes	no
Vinyals et al. [13]	meta-learning + cosine distance classifier	Image	yes	no
Chen et al. [11]	pretrain + finetune	Image	yes	no
Rebuffi et al. [14]	CNN + nearest mean of exemplars classifier	Image	yes	no
Prabhu et al. [15]	CNN + fully connected layer classifier	Image	yes	no
Tao et al. [1]	neural gas network	Image	yes	yes
Zhang et al. [3]	meta-learning + continually evolved classifier	Image	yes	yes
MAPIC	meta-learning + self-attention + distance-based classifier	Time Series	yes	yes

[23], [24], task-incremental learning [25], [26], and domain-incremental learning. Catastrophic forgetting [8] is the main obstacle to incremental learning, in which the model loses its ability to classify old classes when learning the new class. Incremental learning methods fall into three categories according to the way they address catastrophic forgetting, 1) parameter-isolation based [27], 2) replay (or memory)-based [2], [14], [28], [29], 3) regularization-based [2], [14], [23], [25].

A parameter-isolation-based method such as [27], learns the weights of the weights. That means, those parameters are very important for the old classes, so when learning new classes to update parameters, those important parameters are updated with little weight, or even frozen.

Replay-based method not only uses the data of the current class, but also stores part of the data of the old class. [14] designs an exemplar selection strategy to make the selected exemplars closer to the class mean. [15] adopts a greedy balancing sampler to store as much data as memory allows, achieving competitive performance through retraining from scratch.

Regularization-based method mitigates the abrupt changes to the existing classes when the model learns new classes. In incremental learning, knowledge distillation is commonly employed to transfer old model knowledge to the new model. The previous model's output probability is utilized as guidance for the new model.

C. Few-Shot Class-Incremental Learning

Few-shot class-incremental learning is proposed on the basis of CIL, and its change is that there is scarce new class data at incremental stages. To a certain extent, FSCIL is similar to the FL, except that the former only requires the model to classify all the classes it has seen, while the latter requires the model to classify newly emerging classes. [1], [30], [31] put forward FSCIL in image area. [1] uses a neural gas (NG) network to preserve the topology of each class. [31] contradicts the current strategy that allows the updated model to be more similar to the old one. It proposes compressing the embedding of classes that have been seen and reserved for new ones by virtual prototypes.

D. Classification of Physiological Signals

In recent years, the classification of physiological signals using deep learning has been the subject of a great deal of research due to the rapid advancement of artificial intelligence. These researches mainly focus on the classification of

Electrocardiogram and Electroencephalogram (EEG) signals. Both ECG and EEG signals require data pre-processing, that is, removing noise through various filtering methods. The classification of ECG is helpful to the diagnosis of heart disease, and accurate detection of arrhythmia can enable doctors to quickly choose treatment for patients. [5] uses various Machine Learning techniques to classify the ECG signal, including SVM, Adaboost, ANN, and Naïve Bayes. [32] adopts a convolutional neural network (CNN) for classification, and combines generative adaptive networks (GAN) to restore the balance of the dataset. [33] further uses the transformer [34] model for classification. [35] and [36] use AutoEncoder and Long Short-Term Memory neural network(LSTM) to classify EEG, respectively.

The comparison between MAPIC and related works is summarized in Table. I. It is evident that all the existing works are focused on image classification. According to our knowledge, we are the first to investigate FSCIL for medical time series classification. Although [1] and [3] are also applicable to FSCIL, the experiments prove that MAPIC achieves the best performance.

III. PROBLEM DESCRIMINATION

FSCIL includes a base stage and several incremental stages. In the base stage, there is sufficient training data for each class. In an incremental stage, the amount of training data for new classes is small. The trained model should be capable of classifying all classes it has seen. We regard FSCIL in each stage as a task, $\{T_0, T_1, \dots, T_n\}$. The classes to be classified for each task can be seen as $\{C_0, C_1 \dots C_n\}$. T_0 represents the base task, and $\{T_1 \dots T_n\}$ represent n incremental tasks. ΔC_i represents the new classes in the i-th task, $\forall i \neq j, \Delta C_i \cap \Delta C_j = \emptyset, C_0 = \Delta C_0, C_t = \sum_{i=1}^t C_0 + \Delta C_i (t = 1, 2 \dots n)$. $\{D_{train}^0, D_{train}^1, \dots, D_{train}^n\}$ correspond to the train set of $\{T_0, T_1 \dots T_n\}$, respectively. $\{D_{test}^0, D_{test}^1 \dots D_{test}^n\}$ correspond to the test set of $\{T_0, T_1 \dots T_n\}$. We use the episode meta-learning paradigm to solve the few-shot scenario encountered in incremental tasks. D_{train}^i contains a support set S_{train}^i and a query set $Q_{train}^i, i \in [0, n]$. Similarly, $D_{test}^i = S_{test}^i \cup Q_{test}^i$. Metric learning is used to predict the labels of samples. In the prediction process, $p_i, i \in [0, n]$ represents the prototypes of all classes seen in i-th task. Table. II displays the primary notations used in the paper.

IV. META SELF-ATTENTION PROTOTYPE INCREMENTER

We first provide an outline of the suggested MAPIC, as depicted in Fig. 2. It consists of three modules, embedding en-

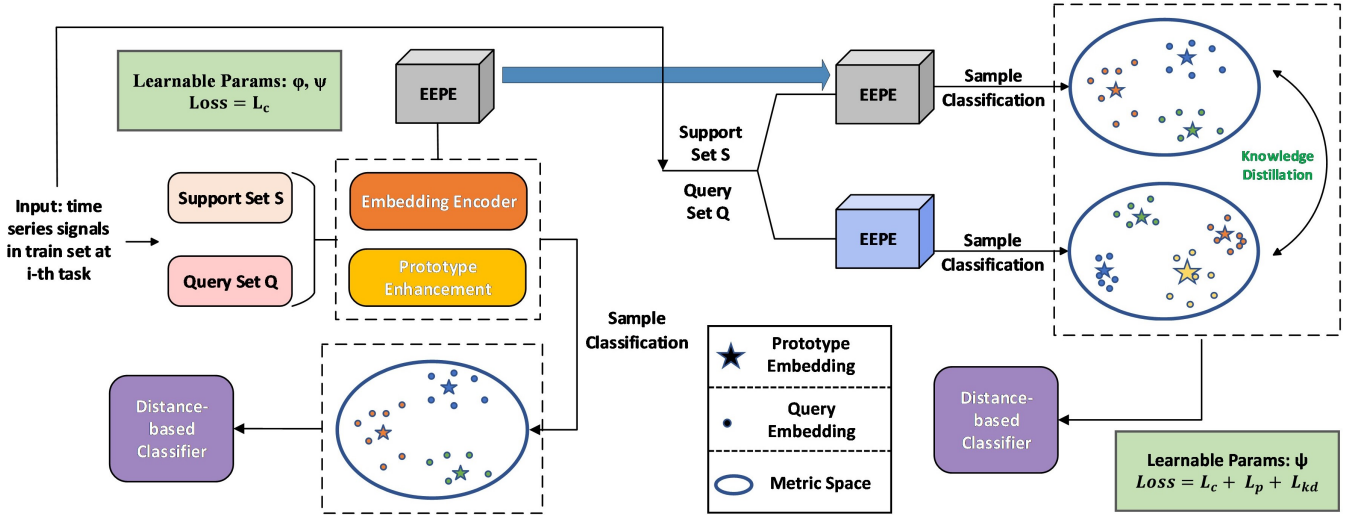


Fig. 2. Outline of the suggested MAPIC for Few-Shot Class-Incremental Learning for time series classification.

TABLE II
MEANING OF THE MAIN NOTATIONS

Notation	Meaning
φ	learnable parameters in embedding encoder module
ψ	learnable parameters in prototype enhancement module
D_{train}^i, D_{test}^i	train set and test set in i-th session
S_{train}^i, Q_{train}^i	support set and query set in i-th task when training
S_{test}^i, Q_{test}^i	support set and query set in i-th task when testing
p_i	prototypes of class i
ΔC_i	new classes in i-th incremental session
C_i	all classes have seen in i-th incremental session
d_k	the dimension of the last layer of embedding encoder
W_Q, W_K, W_V	three different linear transformations in prototype enhancement module
α	correlation coefficient matrix between prototypes
T	temperature hyperparameter in knowledge distillation loss

coder module, prototype enhancement module, and distance-based classifier. The embedding encoder module is used to project one-dimensional time series to high-dimensional feature embedding. We design a prototype enhancement module to get more expressive embedding upon the self-attention mechanism. The distance-based classifier is the nearest neighbour classifier which compares two samples to predict its label. Algorithm 1 shows the MAPIC pseudo code in detail. In the base stage (lines 2-13), we train the parameters of the first two modules, and in each subsequent incremental stage (lines 15-26), we fix the embedding encoder module (line 16) and only train the prototype enhancement module (line 17). The distance-based classifier is non-parametric, and we use the nearest neighbor classifier to make predictions (line 10 and 23). The degree of similarity between the sample and the prototype is determined by the Euclidean distance. This similarity can be seen as a prediction of the sample belonging to the class. We can use it to calculate the loss functions

in the training phase (lines 11 and 24) and classify them in the testing phase. Note that the distance-based classifier is non-parametric, but the process of learning the embedding representations of prototypes and samples is parameterized.

A. Embedding encoder module

The embedding encoder module is designed to encode the support set and query set. The input of this module is one-dimensional time-series signals, and the output is high-dimensional feature embedding. More information is contained in the latter than in the former. When a new class emerges, the difficulty we confront with class-incremental learning is that we must retrain the network so that the model can accommodate the new class. However, the model may suffer from catastrophic forgetting. For purpose of maintaining the model's capacity to classify old classes in incremental stages, we adopt a parameter protection strategy. The embedding encoder trained by the samples in the base stage will be frozen, and in any incremental stage, we use the same frozen embedding encoder for all training samples to obtain their corresponding high-dimensional feature vectors. Fig. 3 depicts the embedding encoder's structural layout.

B. Prototype enhancement module

We adopt an episode meta-learning paradigm to cope with the situation where there are few new class data at incremental stages. Each episode in D_{train}^0 has a support set and a query set, and samples are chosen for each class as training data. Using the embedding encoder module, a corresponding representation of each support set sample is obtained. The prototype of a class is obtained by averaging all the representations of samples in each class in the support set. Equation (1) shows the prototype p_i of the i-th class.

$$p_i = \frac{1}{|M|} \sum_{j=1}^M \varphi(x_i^j) \quad (1)$$

Algorithm 1 MAPIC for few-shot class-incremental learning. φ means the parameters in the embedding encoder module. ψ represents the parameters in the prototype enhancement module. y_{true} , y_{pred} represent the ground truth label and the prediction label of the model respectively.

- 1: **Base task:**
- 2: **Input:** Dataset D_{train}^0 in base task.
- 3: **Require:** φ and ψ of random initialization parameters.
- 4: **Output:** The trained φ and ψ .
- 5: **for** class c **in** D_{train}^0 **do:**
- 6: $\{S_{train}^0, Q_{train}^0\} \leftarrow$ Create the support and query set for every class in D_{train}^0
- 7: $\{p_0, Q_{train}^0\} \leftarrow$ Get prototypes and query embedding of each class upon φ with $\{S_{train}^0, Q_{train}^0\}$
- 8: $p_0' \leftarrow$ Augment p_0 using ψ
- 9: $Q_{train}^0' \leftarrow$ Obtain query embedding in the same metric space as the prototype using ψ
- 10: $y_{pred} \leftarrow$ Make predictions for $\{p_0', Q_{train}^0'\}$ using nearest neighbor classification
- 11: Compute loss $L_c(y_{true}, y_{pred})$
- 12: $\varphi, \psi \leftarrow$ Optimize φ, ψ based on L_c and SGD
- 13: **end for**
- 14: **Incremental tasks:**
- 15: **Input:** Dataset D_{train}^i in incremental tasks.
- 16: **Require:** The frozen φ .
- 17: **Output:** The trained ψ .
- 18: **for** class c **in** D_{train}^i **do:**
- 19: $\{S_{train}^i, Q_{train}^i\} \leftarrow$ Create the support and query set for old classes in D_{train}^i and new class is used in both support set and query set
- 20: $\{p_i, Q_{train}^i\} \leftarrow$ Get prototypes and query embedding of each class upon φ with $\{S_{train}^i, Q_{train}^i\}$
- 21: $p_i' \leftarrow$ Enhance p_i using ψ
- 22: $Q_{train}^i' \leftarrow$ Obtain query embedding in the same metric space as the prototype using ψ
- 23: $y_{pred} \leftarrow$ Make predictions for $\{p_i', Q_{train}^i'\}$ using nearest neighbor classification
- 24: Compute loss $L_{all}(y_{true}, y_{pred})$
- 25: $\psi \leftarrow$ Optimize ψ based on L_{all} and SGD
- 26: **end for**

where M represents all the data in the support set of class i and φ represents the parameters in the embedding encoder. x_i^j is all the samples belonging to i -th class and p_i is the prototype of class i .

There exists some relationship between different prototypes. However, the prototype calculated by (1) cannot capture the information of inter-class relationships. We use a self-attention mechanism to enhance the prototype to capture such related information. We take the prototype of each class as input for the self-attention model. By using self-attention, each prototype can attend to information from different prototypes at different positions. Especially, for those new classes with few data, the prototypes usually contain a significant amount of noise. We can reduce it by information aggregation and

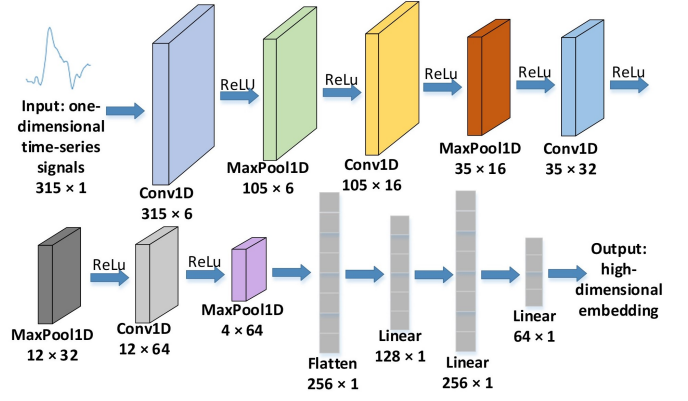


Fig. 3. The structure of embedding encoder module.

enable the prototypes to be more expressive.

Fig. 4 shows the prototype enhancement module. In the self-attention mechanism, the prototype relationship is obtained by multiplying the matrices Query ($Q = W_Q \cdot p_i$) and Key ($K = W_K \cdot p_j$), which is a correlation coefficient matrix. Equation (2) shows the coefficient of the i -th and j -th prototypes.

$$\alpha_{i,j} = \text{softmax}\left(\frac{[W_Q(p_i)] \cdot [W_K(p_j)]^T}{\sqrt{d_k}}\right) \quad (2)$$

where p_i and p_j mean the i -th and j -th class prototypes, respectively. W_Q and W_K are different linear transformations. T is the transpose operation, and d_k is the dimension of the prototype. α is the correlation coefficient matrix.

We update each prototype based on the correlation coefficient matrix. By feeding the prototype into the self-attention mechanism, we can get the augmented prototype. Equation (3) shows the enhanced prototype of the i -th class.

$$p_i' = p_i + W_V(p_i) \cdot \alpha_{i,j} \quad (3)$$

where p_i' is the new prototype after updating and W_V is a linear transformation.

C. Distance-based classifier

Samples are classified based on their distances from prototypes. A sample belongs to the class whose prototype is closest to it. Metric learning requires that the two being compared must belong to the same feature space. Therefore, the samples in the query set must first pass through the embedding encoder module and prototype enhancement module. After prediction, we define three loss functions and update the model using various loss function combinations at various stages.

1) **Sample classification loss:** This loss function measures how well a sample is correctly classified. The distance between each sample and the prototype is determined, and the prototype closest to the sample is selected as its label. Typically, we use a regular cross-entropy loss function to measure how good the classification strategy is. However, unlike regular cross-entropy, we use the distance metric to represent the closeness to each class rather than the probability of the network's output. Since the distances are all positive, and the smallest distance represents the most likely class. However, in the

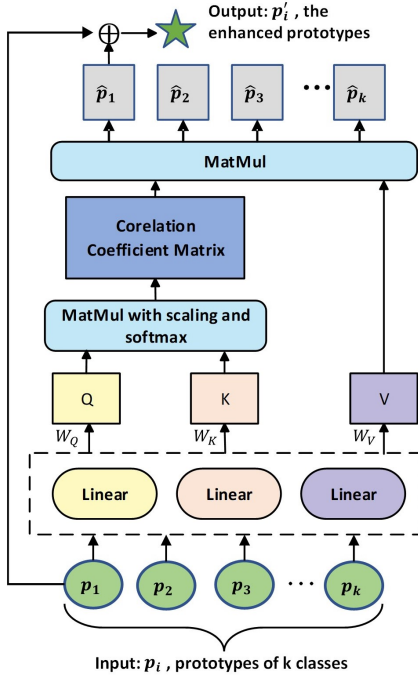


Fig. 4. The process of prototype enhancement module.

softmax step of the cross entropy function, the class to which it belongs has the highest probability. So, we take the negative of distances so that the score obtained by negative distances after softmax is the largest.

$$\text{dist}(x_i, p_j) = \sqrt{(x_i - p_j)^2} \quad (4)$$

$$L_c = -\frac{1}{|N|} \sum_{i=1}^N Y \cdot \log \frac{\exp(-\text{dist}(x_i, p_j))}{\sum_{j=1}^k \exp(-\text{dist}(x_i, p_j))} \quad (5)$$

where Y is the ground truth, x_i is the samples in the query set, and N indicates the quantity of samples in the query set. p_j means the prototypes that have been seen and k represents the number of prototypes.

2) Prototype non-overlapping loss: As new classes emerge during the incremental stages, their prototypes may overlap in the metric space with those of old classes. Once overlap occurs, the classification capacity of the model based on distance will be drastically diminished. Therefore, at incremental stages, we should keep the prototype of a new class as far as possible from the prototypes of the old classes. The model can have a high degree of discrimination between the old and new classes. We measure the distance between the old prototypes and the new prototype using the Euclidean distance. We employ the nonlinear monotonic decreasing function as the loss function, with the closest distance between prototypes serving as the independent variable. Consequently, when the loss is optimized, the minimum distance between prototypes will increase, followed by an increase in all distances.

$$L_p = \frac{1}{|C_i|} \sum_{i \in C_i, j \in C_{i-1}} \min(\exp(-\text{dist}(p_i, p_j))) \quad (6)$$

where C_{i-1} indicates the number of classes in (i-1)-th stage and C_i represents the quantity of all classes in i-th stage.

3) Knowledge distillation loss: In class incremental learning, in addition to adopting replay-based methods to combat catastrophic forgetting, regularization-based methods are also used. Among the regularization-based methods, the most widely used is knowledge distillation. Knowledge distillation was originally designed for model compression. In order to enable a lightweight model to have the ability of a large model as much as possible, we employ the output of the large model as a soft label to guide the lightweight model. In FSCIL, we regard the model in the last stage as the teacher model and the model in the current stage as the student model. We calculate the distance to get the predictions, and divide it by the temperature hyper-parameter. Then we use the softmax function to get the prediction probability, which is used as the soft label. The maximum value in the soft label still represents the class of the sample, and other values can reflect the relationship between different classes. Equation (7) and equation (8) are used to get soft labels in the last and current stages. The teacher model's soft labels only contain the classes it has seen. Therefore, in order to preserve the memory of the old classes of the student model, Equation (9) compares the similarity of those classes that the teacher model has seen in the last stage. As the soft label has more information than the ordinary one-hot label, the output of the student model should be not only close to the ground truth labels, but also as close as possible to the teacher model's soft labels. By equation(7) and equation (8), we get two sets of probability scores. We assess the approximation between these two sets of probability distributions using Kullback-Leibler divergence. By optimizing KL divergence, the output of the student model approaches that of the teacher model as closely as possible, so as to transfer knowledge between stages. Through knowledge distillation, it is ensured that the model still has the ability to classify old classes at incremental stages.

$$\widehat{o'_{tea}} = \frac{\exp(o_{tea}^i/T)}{\sum_{j=1}^n \exp(o_{tea}^j/T)} \quad (7)$$

$$\widehat{o'_{stu}} = \frac{\exp(o_{stu}^i/T)}{\sum_{j=1}^n \exp(o_{stu}^j/T)} \quad (8)$$

$$L_{kd}(\widehat{o'_{stu}}, \widehat{o'_{tea}}) = \sum_{i=1}^{C_{i-1}} \widehat{o'_{stu}}^{(i)} \cdot \log \frac{\widehat{o'_{stu}}^{(i)}}{\widehat{o'_{tea}}^{(i)}} \quad (9)$$

where o_{tea}^i and o_{stu}^i represent the output of the teacher model and student model, respectively. T is the temperature hyperparameter, and C_{i-1} means the number of classes in (i-1)-th stage.

In the base stage, we choose (5) as the loss function, where the parameters we need to update are those in the embedding encoder and self-attention mechanism.

$$L_{all} = L_c + L_p + L_{kd} \quad (10)$$

At incremental stages, we use (10) as the loss function, where we fix the parameters of the embedding encoder and only update the parameters in the prototype enhancement module.

V. EXPERIMENTS

A. Implementation Details

Our implementation is based on Pytorch, and all models are trained on Nvidia RTX 3090. We construct three 1-D ConvNet with different layers plus several fully connected layers to form the backbone, which are used as feature encoders for three datasets. When testing, we use Euclidean distance as the classification standard instead of a fully connected layer classifier. We select the same optimizer and set the same learning rate on Mit-BIH, FaceAll, and UWave datasets. A stochastic gradient descent optimizer is utilized with a fixed learning rate of 0.1 at base stage and 0.01 at the incremental stages. When using knowledge distillation loss, the temperature hyper-parameter is set to 2. At the base stage, we set 200, 18, and 400 episodes to imitate few-shot scenarios on Mit-BIH, FaceAll, and UWave dataset, respectively. We evaluate the performance of MAPIC in the following aspects: comparing with state-of-the-art methods, ablation study, the influence of key parameters, and visualization of the classification effect.

Evaluation Protocol: We record the accuracy of the model at each stage as a measure of the classification ability of the model at the current stage. We also define a performance dropping rate (PD) that measures the absolute accuracy drops at the last stage w.r.t. the accuracy at the base stage, *i.e.*, $PD = Acc^0 - Acc^N$ where Acc^0 is the classification accuracy at the base stage and Acc^N is the accuracy at the last stage. In order to reflect the change of the classification ability of the model, we define a new metric: Relative Performance Dropping rate (RPD), which is the PD divided by the accuracy of the base stage, *i.e.*, $RPD = \frac{PD}{Acc^0}$.

B. Preprocessing of Datasets

Mit-BIH. Mit-BIH [37] is a one-dimensional ECG signal with a sampling frequency of 360 Hz. We take 150 points before and after the R peak as a heartbeat. We set 5 classes as base classes and 4 classes as incremental classes. The 4 new classes are further subdivided into 4 stages, which are subject to the 1-way-5-shot setting.

FaceAll. We select FaceAll dataset from UCR Time Series Classification Archive [38], which contains 14 classes in total. Each sample contains 131 points. We set 5 classes as base classes and 9 classes as incremental classes. The 9 new classes are further subdivided into 9 stages, which are subject to the 1-way-3-shot setting.

UWave. We selected UWaveGestureLibraryX, UWaveGestureLibraryY and UWaveGestureLibraryZ from the UCR Time Series Classification Archive [38]. Each of them has 8 classes, and we combine them to build a composite dataset of 24 classes. There are 315 points in each sample. We set 12 classes as base classes and 12 classes as incremental classes. The 12 new classes are further subdivided into 4 stages, which are subject to the 3-way-5-shot setting.

Mit-BIH Long-Term ECG. Mit-BIH Long-Term ECG contains 7 long-term ECG recordings (14 to 22 hours each). Its frequency is 128Hz, and we take 49 points before the R peak and 51 points after the R peak as one heartbeat. With the exception of 6 classes, the number of samples of other

classes is close to zero. We set 3 classes as base classes and 3 classes as incremental classes. The 3 new classes are further subdivided into 3 stages, which are subject to the 1-way-5-shot setting.

C. Baseline Methods

We compare the proposed method with following baselines:

- **finetune:** At incremental stages, it applies cross-entropy loss and employs solely new class data for FSCIL.
- **iCaRL:** iCaRL [14] is a class-incremental image classification technique. It constructs an exemplar set, which is not only used as training data in the training stage, but also used as getting prototype for classification in the testing stage. In iCaRL, the quantity of samples of each class is equal to the memory size divided by the number of classes that have ever been seen. However, during the training process in FSCIL, the number of examples of new classes is negligible.
- **Gdumb [15]:** It greedily stores all samples as long as the memory allows. In the experiment, we do not set the memory size and let it use all the data encountered in the current stage for training. By changing the mask matrix, task-incremental learning, and class-incremental learning can be done.
- **CEC:** CEC [3] has a pseudo incremental learning process between the base stage and incremental stages. In this process, it is necessary to make some changes to the old class samples to synchronize the new pseudo incremental classes. As we conduct experiments on one-dimensional time series, we add random noise to synthesize new pseudo-incremental classes.

It is vital to note that each of the aforementioned approaches utilizes the same backbone on the same dataset.

D. Performance Comparison

MAPIC and other baselines are executed five times with various random seeds. We keep track of the best outcome at each stage of each trial and summarize the average test precision at each incremental stage in Table. III, Table. IV, Table. V and Fig. 5.

From Tables. III to V, MAPIC is superior to other baselines in the three datasets of Mit-BIH, FaceAll, and UWave. Results indicate that MAPIC achieves the highest performance in nearly every stage. It gets 27.99%, 18.4% and 3.95% performance improvement over the best baseline model (CEC, iCaRL, CEC) in the last incremental stage on Mit-BIH, FaceAll and UWave, respectively. MAPIC achieves the lowest PD and RPD compared with other methods on the Mit-BIH dataset, suggesting that MAPIC is the most resistant to catastrophic forgetting. Although iCaRL achieves the lowest PD and RPD on the UWave dataset, its accuracy at the initial stage is far lower than that of MAPIC. Similarly, MAPIC achieves the lowest RD and RPD except for the iCaRL method.

From Fig. 5, due to the insufficient training data of new classes available, the classification accuracy on all classes continues declining. MAPIC has the most gentle downward

TABLE III
COMPARISON WITH THE BASELINES ON MIT-BIH DATASET

Method	Acc. In each stage(%)						
	Base	Stage 1	Stage 2	Stage 3	Stage 4	PD ↓	RPD ↓
finetune	98.67±0.67%	68.22±17.91%	37.64±15.0%	30.08±24.52%	24.72±13.5%	73.95%	74.94%
iCaRL	88.2±4.47%	83.41±13.0%	73.22±11.2%	58.61±9.19%	42.82±6.7%	45.38%	51.45%
Gdumb	98.55±0.58%	69.66±0.54%	51.57±0.82%	41.86±0.54%	33.7±1.17%	64.85%	65.8%
CEC	98.68±0.6%	93.52±3.51%	79.74±5.37%	65.95±6.95%	54.56±5.45%	44.12%	44.71%
MAPIC(Ours)	99.22±0.25%	98.84±0.44%	92.27±1.36%	86.89±3.31%	82.55±3.14%	16.67%	16.8%

TABLE IV
COMPARISON WITH THE BASELINES ON FACEALL DATASET

Method	Acc. In each stage(%)											
	Base	Stage 1	Stage 2	Stage 3	Stage 4	Stage 5	Stage 6	Stage 7	Stage 8	Stage 9	PD ↓	RPD ↓
finetune	82.23±4.53%	18.38±12.47%	15.83±18.13%	12.12±12.83%	6.34±2.52%	2.0±0.0%	1.6±0.0%	1.52±0.0%	1.9±0.35%	1.84±0.54%	80.39%	97.77%
iCaRL	78.57±4.51%	68.63±3.19%	63.26±7.36%	57.74±6.28%	54.7±5.38%	52.66±3.88%	47.33±4.46%	40.34±3.11%	35.97±2.63%	30.08±3.07%	48.49%	61.72%
Gdumb	72.76±19.64%	43.69±17.68%	27.92±2.02%	19.36±2.07%	17.82±1.75%	11.12±1.84%	12.7±3.5%	8.47±1.83%	5.55±1.84%	5.44±2.84%	67.32%	92.52%
CEC	88.95±6.88%	71.6±3.59%	62.49±5.37%	51.57±7.96%	52.15±4.5%	47.49±8.15%	40.71±5.69%	34.83±4.51%	28.38±3.8%	24.49±4.44%	64.46%	72.47%
MAPIC(Ours)	98.54±0.42%	92.22±0.21%	85.17±0.36%	78.99±3.11%	74.26±2.19%	69.46±2.61%	64.83±1.78%	56.49±1.77%	51.33±1.49%	48.48±0.99%	50.06%	50.8%

TABLE V
COMPARISON WITH THE BASELINES ON UWAVE DATASET

Method	Acc. In each stage(%)						
	Base	Stage 1	Stage 2	Stage 3	Stage 4	PD ↓	RPD ↓
finetune	65.59±2.61%	35.07±15.62%	25.48±2.94%	9.07±0.49%	6.8±2.56%	58.79%	89.63%
iCaRL	42.63±2.61%	34.66±2.47%	31.89±3.02%	24.68±1.47%	21.82±1.01%	20.81%	48.81%
Gdumb	52.87±9.4%	27.95±2.18%	16.8±1.13%	12.32±3.32%	8.22±2.52%	44.65%	84.45%
CEC	55.63±6.99%	40.69±3.08%	37.99±3.21%	30.28±2.98%	26.73±2.15%	28.9%	51.05%
MAPIC(Ours)	63.55±2.5%	48.71±2.54%	41.35±3.06%	34.68±3.65%	30.68±2.62%	32.87%	51.72%

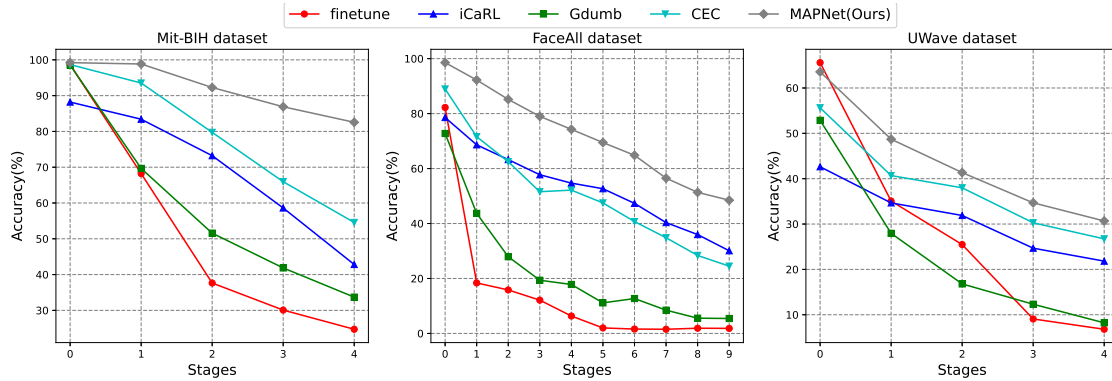


Fig. 5. Comparison on three datasets. MAPIC has significant performance benefits over prior works.

TABLE VI
ABLATION STUDY ON FACEALL DATASET

Different Modules				FaceAll (1-way-3-shot)							
PP	L _{kd}	L _p	PE	Base	Stage 1	Stage 2	Stage 3	Stage 4	Stage 5	Stage 6	Stage 7
✓	✓	✓	✓	98.54±0.42%	60.25±15.8%	69.27±21.26%	60.67±15.38%	53.61±5.11%	45.63±19.63%	43.69±18.66%	38.74±14.12%
✓	✓	✓	✓	85.99±1.99%	79.71±1.0%	76.65±0.38%	72.3±1.79%	66.08±1.49%	64.29±1.95%	63.12±1.84%	56.4±0.92%
✓	✓	✓	✓	85.99±1.99%	80.98±1.02%	78.67±1.33%	72.79±2.15%	66.42±1.24%	65.3±2.85%	61.33±3.61%	54.86±3.5%
✓	✓	✓	✓	85.99±1.99%	79.35±1.13%	74.98±0.84%	70.65±1.32%	64.28±1.48%	60.71±4.08%	59.01±2.5%	53.92±0.52%
✓	✓	✓	✓	98.54±0.42%	92.22±0.21%	85.17±0.36%	78.99±3.11%	74.26±2.19%	69.46±2.61%	64.83±1.78%	56.49±1.77%

trend, which proves that it has the best ability to resist catastrophic forgetting. Finetune achieves the worst performance, and accuracy will drop sharply in the first incremental stage. Obviously, it is difficult to finetune to classify all classes by

using only a few samples at incremental stages. What's more, finetune and Gdumb which both use the fully connected layer classifier get poorer performance, while iCaRL, CEC, and MAPIC which are inspired by metric learning, use distance-

TABLE VII
ABLATION STUDY ON UWAVE DATASET

Different Modules				Uwave (3-way-5-shot)				
PP	L_kd	L_p	PE	Base	Stage 1	Stage 2	Stage 3	Stage 4
	✓	✓	✓	63.55±2.5%	28.58±1.99%	25.93±5.35%	20.27±0.78%	20.98±3.34%
✓			✓	63.55±2.5%	48.62±2.58%	40.98±3.1%	33.83±2.48%	30.66±2.36%
✓	✓		✓	63.55±2.5%	48.31±2.9%	42.01±3.26%	33.52±3.23%	29.61±2.93%
✓	✓			50.76±2.14%	43.66±4.14%	41.54±3.99%	32.96±3.15%	30.61±2.58%
✓	✓	✓	✓	63.55±2.5%	48.71±2.54%	41.35±3.06%	34.68±3.65%	30.68±2.62%

TABLE VIII
ABLATION STUDY ON MIT-BIH LONG-TERM ECG DATASET

Different Modules				Mit-BIH Long-Term ECG (1-way-5-shot)			
PP	L_kd	L_p	PE	Base	Stage 1	Stage 2	Stage 3
	✓	✓	✓	99.89±0.08%	30.58±39.79%	20.98±28.06%	18.78±25.59%
✓			✓	99.89±0.08%	99.01±0.52%	89.41±5.98%	87.72±4.43%
✓	✓		✓	99.89±0.08%	31.9±41.67%	28.82±39.14%	27.4±37.79%
✓	✓	✓		99.45±0.21%	95.67±3.15%	81.26±6.69%	78.2±4.56%
✓	✓	✓	✓	99.89±0.08%	98.86±0.67%	89.28±4.84%	88.6±4.66%

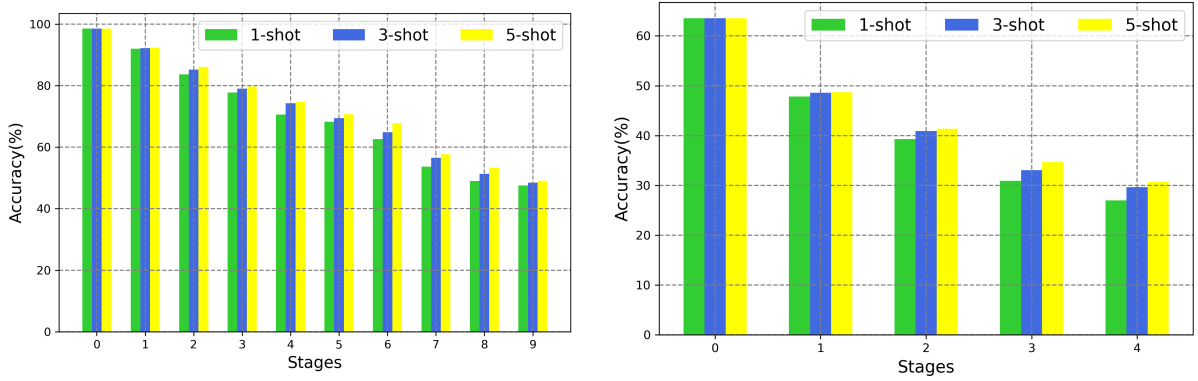


Fig. 6. Parameter Analysis of the number of samples for novel classes on FaceAll and UWave dataset.

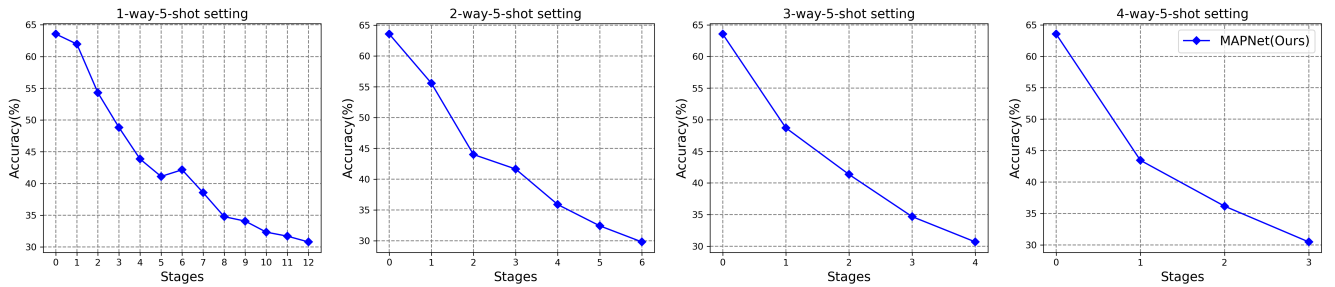


Fig. 7. Observe how the accuracy changes as the number of new classes in each incremental stage is altered.

based classifiers, and get better performance. MAPIC makes predictions based on the Euclidean distance between the prototypes and query embedding. Training with the distance-based classifier can explicitly reduce intra-class variations. Compared with iCaRL and CEC, we adopt the self-attention mechanism to enhance the prototypes, which can capture the inter-class relationships and make prototypes more representative. The parameter protection strategy and knowledge distillation also preserve MAPIC's ability to classify old classes.

E. Ablation Study

In this section, we undertake an ablative analysis based on FaceAll, UWave, and Mit-BIH Long-Term ECG datasets to evaluate the efficacy of our model's various components. The experimental outcomes are shown in Table. VI, Table. VII, and Table. VIII.

- **No parameter protection(PP):** MAPIC contains an embedding encoder module and a prototype enhancement module. In the base stage, the model trains both their

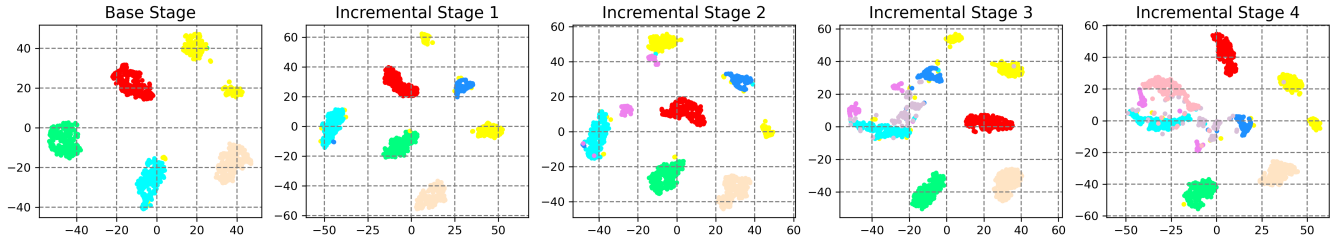


Fig. 8. t-SNE visualization of classification effect upon MAPIC in each stage on Mit-BIH dataset. Different colored dots represent data points from distinct classes.

parameters. In the incremental stages, the embedding encoder is frozen, and only the parameters in the prototype enhancement module will be trained.

- **No prototype enhancement:** In the base and incremental stages, the prototype enhancement module is not utilized, and only the encoder's output is utilized for representation learning and classification.
- **No loss functions(L_{kd}, L_p):** In the incremental stages, there are three loss functions in total. We compare the knowledge distillation loss and prototype non-overlapping loss.

From Table. VI, we can find that MAPIC achieves the highest accuracy among all incremental stages. The accuracy is down by 12.51% at most without a parameter protection strategy. The prototype enhancement module increases the expressiveness of the prototypes and improves the precision accuracy at every incremental stage. We examine various loss function combinations and find that eliminating any loss function will have a substantial effect on performance.

From Table. VII, we can find that MAPIC almost achieves the highest accuracy among all incremental stages. Without a parameter protection strategy, the accuracy at each incremental stage drops significantly. The prototype enhancement module improves the accuracy by up to 12.79%. The prototype non-overlapping loss separates the prototypes so that the classification is clearer. Knowledge distillation loss also retains the ability to classify old classes to resist catastrophic forgetting.

From Table. VIII, we can find that MAPIC almost achieves the highest accuracy among all incremental stages. The accuracy is down by 69.82% at the third stage without a parameter protection strategy. Without the prototype enhancement module, the accuracy is lower than MAPIC at each stage. Though the method without knowledge distillation loss achieves the highest accuracy at stage 2, knowledge distillation can maintain a higher classification accuracy rate in the long run.

F. Influence of Parameter

On the FaceAll dataset, we adopt a 1-way-3-shot setting at incremental stages. On the UWave dataset, we adopt a 3-way-5-shot setting at incremental stages. The experience of few-shot learning indicates that the classification impact improves with the number of samples for the new class. We compare 1-shot, 3-shot, and 5-shot on these two datasets. The experimental outcomes are depicted in Fig. 6. As the quantity of new class samples increases, it is evident that the

5-shot performs the best and the 1-shot performs the worst. In accordance with the properties of deep learning, classification performance improves as the amount of training data increases.

On the UWave dataset, we observe changes in accuracy by changing the number of incremental classes at incremental stages. We set each incremental stage to 1, 2, 3, and 4 classes. From Fig. 7, we can observe that the first stage's accuracy decreases as the number of new classes rise although there are fewer incremental steps. It can also be shown that the more new classes are added, the more difficult for a model to learn. Intuitively, the more incremental steps, the more frequently catastrophic forgetting occurs. However, in our method, with different incremental steps, their final accuracy is almost the same, which indicates that our model is robust to catastrophic forgetting.

G. Visualization of classification effect

As illustrated in Fig. 8, we utilize the t-SNE approach to project the representations of test data at the base stage and four incremental stages onto the Mit-BIH dataset. As new classes are added, samples from the same class will typically be well clustered, while samples from different classes would typically be dispersed.

H. Case Study

MAPIC is an outstanding framework for medical time series classification. We can deploy MAPIC on smart devices, such as watches and bracelets. These devices can monitor physiological signals, and MAPIC can be used to detect abnormalities. People wear smart devices to monitor ECG in their lives. Once abnormal heartbeat occurs, MAPIC can learn to classify them based on a small number of abnormal signals and remind the users. That is, MAPIC can classify new classes of heartbeats online.

VI. CONCLUSION

In this study, we propose MAPIC for medical time series classification using few-shot class-incremental learning. MAPIC adopts a parameter protection strategy to prevent the parameters crucial to old classes from updating. The embedding encoder module is like a feature extraction, mapping all samples in the support set and a query set into high-dimensional embeddings with rich information. The distance-based classifier is the nearest neighbor classifier based on

Euclidean distance, which requires that samples and prototypes are in the same metric space. To reduce intra-class variations, we propose a prototype enhancement module to update the prototypes by a self-attention mechanism, making prototypes more expressive. We define three loss functions to train MAPIC. The sample classification loss and prototype non-overlapping loss are used to increase the accuracy. The knowledge distillation loss is utilized to transfer knowledge between stages, thus preserving the model's classification of old classes. Experiments on three different datasets demonstrate that MAPIC performs noticeably better than the state-of-the-art baselines.

In the future, we will improve MAPIC to make it light enough to be deployed on the edge device, so that it can process online FSCIL.

REFERENCES

- [1] X. Tao, X. Hong, X. Chang, S. Dong, X. Wei, and Y. Gong, "Few-shot class-incremental learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 12 183–12 192.
- [2] J. He and F. Zhu, "Online continual learning for visual food classification," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 2337–2346.
- [3] C. Zhang, N. Song, G. Lin, Y. Zheng, P. Pan, and Y. Xu, "Few-shot incremental learning with continually evolved classifiers," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 12 455–12 464.
- [4] L. Sun, Z. Zhong, C. Zhang, Y. Zhang, and D. Wu, "Tess: multivariate sensor time series prediction for building sustainable smart cities," *ACM Transactions on Sensor Networks*, 2022.
- [5] S. Celin and K. Vasanth, "Ecg signal classification using various machine learning techniques," *Journal of medical systems*, vol. 42, no. 12, pp. 1–11, 2018.
- [6] X. Ning, W. Tian, Z. Yu, W. Li, X. Bai, and Y. Wang, "Hcfnn: high-order coverage function neural network for image classification," *Pattern Recognition*, vol. 131, p. 108873, 2022.
- [7] C. Wang, X. Ning, L. Sun, L. Zhang, W. Li, and X. Bai, "Learning discriminative features by covering local geometric space for point cloud analysis," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 60, pp. 1–15, 2022.
- [8] M. McCloskey and N. J. Cohen, "Catastrophic interference in connectionist networks: The sequential learning problem," in *Psychology of learning and motivation*. Elsevier, 1989, vol. 24, pp. 109–165.
- [9] X. Dong, C. Wang, and W. Si, "Ecg beat classification via deterministic learning," *Neurocomputing*, vol. 240, pp. 1–12, 2017.
- [10] B. Lu, X. Gan, L. Yang, W. Zhang, L. Fu, and X. Wang, "Geometer: Graph few-shot class-incremental learning via prototype representation," *arXiv preprint arXiv:2205.13954*, 2022.
- [11] W.-Y. Chen, Y.-C. Liu, Z. Kira, Y.-C. F. Wang, and J.-B. Huang, "A closer look at few-shot classification," *arXiv preprint arXiv:1904.04232*, 2019.
- [12] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," *Advances in neural information processing systems*, vol. 30, 2017.
- [13] O. Vinyals, C. Blundell, T. Lillicrap, D. Wierstra *et al.*, "Matching networks for one shot learning," *Advances in neural information processing systems*, vol. 29, 2016.
- [14] S.-A. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert, "icarl: Incremental classifier and representation learning," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2017, pp. 2001–2010.
- [15] A. Prabhu, P. H. Torr, and P. K. Dokania, "Gdumb: A simple approach that questions our progress in continual learning," in *European conference on computer vision*. Springer, 2020, pp. 524–540.
- [16] R. Hou, H. Chang, B. Ma, S. Shan, and X. Chen, "Cross attention network for few-shot classification," *Advances in Neural Information Processing Systems*, vol. 32, 2019.
- [17] G. Koch, R. Zemel, R. Salakhutdinov *et al.*, "Siamese neural networks for one-shot image recognition," in *ICML deep learning workshop*, vol. 2. Lille, 2015, p. 0.
- [18] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *International conference on machine learning*. PMLR, 2017, pp. 1126–1135.
- [19] M. A. Jamal and G.-J. Qi, "Task agnostic meta-learning for few-shot learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 11 719–11 727.
- [20] A. Chaudhry, M. Ranzato, M. Rohrbach, and M. Elhoseiny, "Efficient lifelong learning with a-gem," *arXiv preprint arXiv:1812.00420*, 2018.
- [21] P. Dhar, R. V. Singh, K.-C. Peng, Z. Wu, and R. Chellappa, "Learning without memorizing," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 5138–5146.
- [22] G. M. Van de Ven and A. S. Tolias, "Three scenarios for continual learning," *arXiv preprint arXiv:1904.07734*, 2019.
- [23] F. M. Castro, M. J. Marín-Jiménez, N. Guil, C. Schmid, and K. Alahari, "End-to-end incremental learning," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 233–248.
- [24] S. Hou, X. Pan, C. C. Loy, Z. Wang, and D. Lin, "Learning a unified classifier incrementally via rebalancing," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 831–839.
- [25] Z. Li and D. Hoiem, "Learning without forgetting," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 12, pp. 2935–2947, 2017.
- [26] L. Sun and J. Wu, "A scalable and transferable federated learning system for classifying healthcare sensor data," *IEEE Journal of Biomedical and Health Informatics*, 2022.
- [27] R. Aljundi, F. Babiloni, M. Elhoseiny, M. Rohrbach, and T. Tuytelaars, "Memory aware synapses: Learning what (not) to forget," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 139–154.
- [28] D. Lopez-Paz and M. Ranzato, "Gradient episodic memory for continual learning," *Advances in neural information processing systems*, vol. 30, 2017.
- [29] D. Rolnick, A. Ahuja, J. Schwarz, T. Lillicrap, and G. Wayne, "Experience replay for continual learning," *Advances in Neural Information Processing Systems*, vol. 32, 2019.
- [30] A. Cheraghian, S. Rahman, P. Fang, S. K. Roy, L. Petersson, and M. Harandi, "Semantic-aware knowledge distillation for few-shot class-incremental learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 2534–2543.
- [31] D.-W. Zhou, F.-Y. Wang, H.-J. Ye, L. Ma, S. Pu, and D.-C. Zhan, "Forward compatible few-shot class-incremental learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 9046–9056.
- [32] A. M. Shaker, M. Tantawi, H. A. Shedeed, and M. F. Tolba, "Generalization of convolutional neural networks for ecg classification using generative adversarial networks," *IEEE Access*, vol. 8, pp. 35 592–35 605, 2020.
- [33] G. Yan, S. Liang, Y. Zhang, and F. Liu, "Fusing transformer model with temporal features for ecg heartbeat classification," in *2019 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*. IEEE, 2019, pp. 898–905.
- [34] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.
- [35] L. Sun, Z. Zhong, Z. Qu, and N. Xiong, "Perae: an effective personalized autoencoder for ecg-based biometric in augmented reality system," *IEEE journal of biomedical and health informatics*, vol. 26, no. 6, pp. 2435–2446, 2022.
- [36] P. Nagabushanam, S. Thomas George, and S. Radha, "Eeg signal classification using lstm and improved neural network algorithms," *Soft Computing*, vol. 24, no. 13, pp. 9981–10003, 2020.
- [37] A. L. Goldberger, L. A. Amaral, L. Glass, J. M. Hausdorff, P. C. Ivanov, R. G. Mark, J. E. Mietus, G. B. Moody, C.-K. Peng, and H. E. Stanley, "Physiobank, physiobank, and physionet: components of a new research resource for complex physiologic signals," *circulation*, vol. 101, no. 23, pp. e215–e220, 2000.
- [38] 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," *IEEE/CAA Journal of Automatica Sinica*, vol. 6, no. 6, pp. 1293–1305, 2019.