iTimes: Investigating Semisupervised Time Series Classification via Irregular Time Sampling

Xuxin Liu, Fengbin Zhang, Han Liu, and Haoyi Fan

***Abstract*—Semi-supervised learning (SSL) provides a powerful paradigm to mitigate the reliance on large labeled data by leveraging unlabeled data during model training. However, for time series data, few SSL models focus on the underlying temporal structure of time series, which results in a suboptimal representation learning quality on unlabeled time series. In this article, we propose a framework of semisupervised time series classification by investigating irregular time sampling (*iTimes*), which learns the underlying temporal structure of unlabeled time series in a self-supervised manner to benefit semisupervised time series classification. Specifically, we propose four different irregular time sampling functions to transform the original time series into different transformations. Then, iTimes employs a supervised module to classify labeled time series directly and employs a self-supervised module on unlabeled time series by predicting the transformation type of irregular time sampling. Finally, the underlying temporal structure pattern of unlabeled time series can be captured in the self-supervised module. The feature spaces between labeled data and unlabeled data can be aligned by jointly training the supervised and self-supervised modules which boost the ability of model learning and the representation quality. Extensive experimental results on multiple real-world datasets demonstrate the effectiveness of iTimes compared with the state-of-the-art baselines.**

***Index Terms*—Irregular time sampling, self-supervised learning, semi-supervised learning (SSL), time series classification.**

I. Introduction

TIME series classification is a fundamental task in many application domains, such as healthcare monitoring [1], [2], activity recognition [3], power system security [4], [5] and fault diagnosis [6]. In recent years, convolutional neural networks (ConvNets) have been extensively used and achieved much success in this field. Specifically, by training the ConvNets using massive manually labeled training data, they managed to learn the representations that facilitate the classification task [6], [7]. However, data annotation is usually expensive and time-consuming, and even impossible in some domains because of expert annotation knowledge.

Due to that, semisupervised, self-supervised learning has gained much attention, which provides a solution to the aforementioned data annotation challenges [6], [7], [8], [9]. Semi-supervised learning (SSL) aims to combine a tremendous amount of unlabeled data with limited labeled data during model training to boost the model's performance. To alleviate the dependence of labeled time series, dynamic time warping (DTW) [10], and maximum diagonal line of cross-recursive quantification analysis [11] distance-based methods are used as classification criteria for one-nearest-neighbor. Different from these measure-based methods, techniques, such as spectral analysis, phase consistency regularization, and adversarial training, are applied in [7] for SSL. Recently, self-supervised learning has achieved great success in natural language processing and computer vision domains, which learns a useful representation of unlabeled data via defining and solving various pretext tasks. Inspired by this, the subsequence forecast is selected as a pretext task in [8] for self-supervised learning on unlabeled time series. Despite the encouraging results of these methods, few of them take into account the underlying temporal structure in the time series, which leaves the representation of learning on unlabeled time series underexplored. More recently, SemiTime [9], takes a past–future relation prediction of time segments as the pretext task, which explores the underlying temporal structure within time series, has made progress. However, the past–future relation prediction tries to capture a coarse-grained temporal structure, which results in a suboptimal representation quality of time series. Therefore, how to design an efficient self-supervised pretext task that captures the temporal structure of unlabeled time series is still an open problem for semisupervised time series classification.

To cope with the aforementioned challenges, in this article, we propose a framework of semisupervised time series classification by investigating irregular time sampling (*iTimes*), which learns the underlying temporal structure of unlabeled time series in a self-supervised manner to benefit semisupervised time series classification. Specifically, as shown in Fig. 1, for labeled data, iTimes classify the time series directly under the

Manuscript received 11 June 2022; accepted 9 August 2022. Date of publication 26 August 2022; date of current version 4 May 2023. Paper no. TII-22-2503. *(Corresponding authors: Fengbin Zhang; Haoyi Fan.)*

Xuxin Liu, Fengbin Zhang, and Han Liu are with the School of Computer Sicence and Technology, Harbin University of Science and Technology, Harbin 150080, China (e-mail: 1909030127@stu.hrbust.edu.cn; zhangfengbin@hrbust.edu.cn; 1820400024@stu.hrbust.edu.cn).

Haoyi Fan is with the School of Computer and Artificial Intelligence, Zhengzhou University, Zhengzhou 450001, China (e-mail: isfanhy@hrbust.edu.cn).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TII.2022.3199374.

Digital Object Identifier 10.1109/TII.2022.3199374

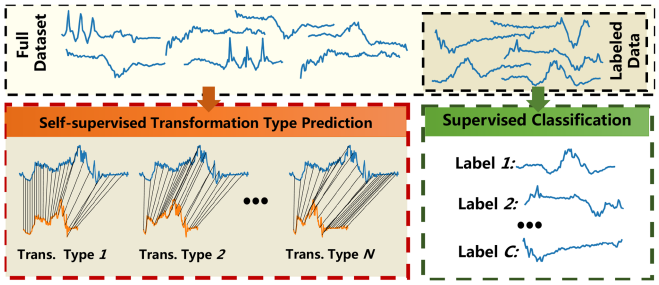

Fig. 1. Schematic diagram of the proposed semisupervised time series classification framework, which conducts the supervised classification on labeled data and conducts the self-supervised transformation (trans.) prediction on unlabeled data.

supervision of the annotated label. For unlabeled data, iTimes follows the self-supervised paradigm, and is proposed to learn the temporal structure by training ConvNets to recognizing the temporal transformations applied to the time series used as input. More specifically, we first define a set of time sampling functions, and then apply these sampling functions to each time series in the dataset. The resulting transformations are then feed to a ConvNets to recognize the transformations of each time series. It is the set time sampling functions that define the classification pretext tasks that iTimes has to learn. Finally, the underlying temporal structure pattern of unlabeled time series can be captured by completing the pretext task in the self-supervised module. The feature spaces between labeled data and unlabeled data can be aligned by jointly training the supervised and self-supervised modules which boost the ability of model learning and the representation quality. Experimental results on multiple real-world datasets of different sources show that iTimes consistently outperforms the state-of-the-art baselines, which demonstrates the effectiveness of the proposed method.

In summary, the main contributions of this article are as follows:

1) We propose a framework of semisupervised time series classification by investigating irregular time sampling (*iTimes*), which learns the underlying temporal structure of unlabeled time series in a self-supervised manner to benefit semisupervised time series classification.
2) We introduce four different irregular time sampling methods to transform the original time series into different transformations, based on which the fine-grained temporal structure can be captured by predicting the types of different transformations.
3) We conduct extensive experiments on multiple real-world time-series datasets, and the results show that our method achieves the state-of-the-art on semisupervised time series classification tasks.

II. Related Work

A. Semisupervised Learning

In the past few years, SSL has gained massive research in the domain of machine learning, which provides an effective way to leverage unlabeled data to improve model performance [9]. Existing SSL approaches can be divided into two categories: 1) entropy minimization-based methods [12], [13] and 2) consistency regularization-based methods [14], [15], [16]. Entropy minimization-based methods assume that the decision boundary should not cross the high-density region of marginal distribution, and, therefore, try to train a classifier that can make low-entropy predictions on unlabeled data. For example, pseudolabel [13] generates pseudolabels of unlabeled data to supervise model training on unlabeled data. Consistency regularization-based methods assume that the model should be invariant to the different input perturbations, e.g., apply different data augmentations to the same input. For example, Π-Model [14] uses self-ensembling to form a consistent prediction of unlabeled data under different augmentation and regularization. MixMatch [15] predicts low-entropy labels for data-augmented unlabeled examples and mixes labeled and unlabeled data using MixUp for training on unlabeled data. FixMatch [16] tries to generate high-confident pseudolabels of the weakly augmented unlabeled input and then use the pseudolabel to supervise the model for learning on the strongly augmented version of the same input. For time series data, DTW distance [10] based methods are used as distance metrics in K-NN classifier to measure the similarity between labeled and unlabeled time series, and a phase-consistency regularization [7] is applied to forbearing time series. More recently, SemiTime [9] takes a past–future relation prediction of time segments as the pretext task to train the model on unlabeled time series. However, the past–future relation prediction tries to capture a coarse-grained temporal structure, which results in a suboptimal representation quality on time series. Therefore, how to design an efficient self-supervised pretext task that captures the temporal structure of unlabeled time series is still an open problem.

B. Self-Supervised Learning

Self-supervised learning aims to learn the underlying structural features within the data under the supervision of the self-generated labels from data, which has achieved great success in natural language processing and computer vision domains by defining and solving various pretext tasks, such as restoring aspect ratio distortion [17], rotation prediction [18], and visual contrastive learning [19] for image, and frame order validation [20], playback rate perception [21] and pace prediction [22] for video. For audio/speech data, there are also different self-supervised techniques, such as multitask learning from raw audio by predicting a number of handcrafted features, such as MFCCs, prosody, and waveform [23], and contrastive learning on audio/speech data [24]. In [25], an inter–intra relational reasoning-based method is proposed to learn the representation of time series by identifying short, medium, and long-term relationships between segments and predicting positive and negative intersample relationships jointly. More recently, in [26], a self-supervised learning model based on the wavelet transform for scalogram-signal correspondence learning is proposed. In [27], self-supervised representation learning using heart rate signals as self-supervised signals for the activity data is proposed. In [28], six different transformations applied to ECG signals are proposed and the recognition of signal transformation types is used as a pretext task.

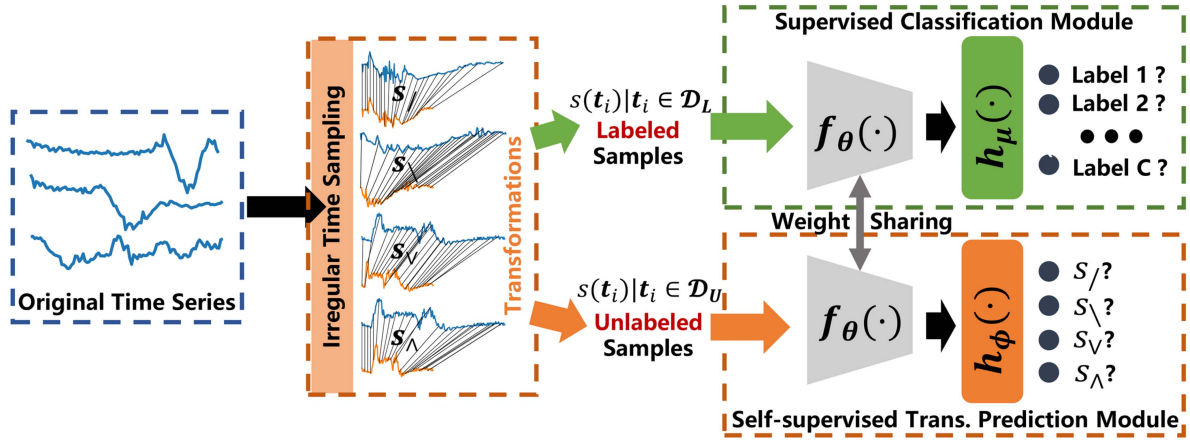

Fig. 2. Architecture of the proposed iTimes.

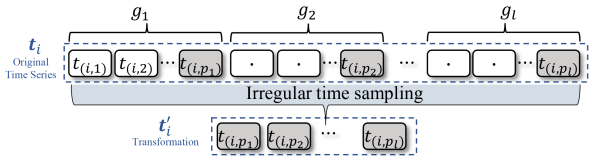

Fig. 3. Schematic diagram of irregular sampling, where p_j is the sampling point, g_j is the time span between p_{j-1} and p_j.

III. Method

In this section, we show the proposed iTimes in detail. As shown in Fig. 2, iTimes consists of three modules, including an irregular time sampling module, a supervised classification module, and a self-supervised transformation prediction module. The inputs of iTimes are a set of labeled input-target pairs $\mathcal{D}_L = \{(\boldsymbol{t}_i, y_i)|\boldsymbol{t}_i = (t_{(i,1)}, \ldots t_{(i,T)}), y_i \in \{1, 2, \ldots, C\}\}_{i=1}^{M}$ and a set of unlabeled T-length time series $\mathcal{D}_U = \{\boldsymbol{t}_i|\boldsymbol{t}_i = (t_{(i,1)}, \ldots t_{(i,T)})\}_{i=1}^{N}$, where $M \ll N$. First, the original time series is transformed into different transformations via irregular time sampling. Then, iTimes conducts the supervised classification directly on the transformations of labeled data $\mathcal{D}_L$, and conducts the self-supervised transformation prediction on the transformations of unlabeled data $\mathcal{D}_U$. In iTimes, a weight sharing backbone encoder f_θ is first employed to extract the feature of transformation inputs, and following which, a classification head h_μ and a prediction head h_φ are used for supervised classification and prediction, respectively.

A. Irregular Time Sampling

In this section, we illustrate the irregular time sampling function in detail, which refers to the sampling where the time span between the sampling points is not fixed. In this work, we use the function $s_\bullet(\boldsymbol{t}_i, \alpha)$ to represent irregular time sampling, which takes the time series $\boldsymbol{t}_i$ and the sampling rate α as inputs to generate an l-length transformed time series $\boldsymbol{t}'_i$, where $l = \lceil T \cdot \alpha \rceil$. As shown in Fig. 3, irregular time sampling first split the time series $\boldsymbol{t}_i = (t_{(i,1)}, \ldots t_{(i,T)})$ into l consecutive time segments $\{\boldsymbol{t}_{i,1}, \boldsymbol{t}_{i,2}, \ldots \boldsymbol{t}_{i,l}\}$, with different time spans $\boldsymbol{g}_i = \{g_1, g_2, \ldots g_l\}$ where $\sum_{j=1}^{l} g_j = T$. Then, the last timestamp $t_{(i,p_j)}$ of each segment $\boldsymbol{t}_{i,j}$ is sampled to construct the transformed time series $\boldsymbol{t}'_i = (t_{(i,p_1)}, t_{(i,p_2)}, \ldots, t_{(i,p_l)})$, $p_j = p_{j-1} + g_j, p_0 = 0$.

Specifically, we define a set of irregular time sampling functions $\mathcal{S} = \{s_/(\boldsymbol{t}_i, \alpha), s_\setminus(\boldsymbol{t}_i, \alpha), s_\vee(\boldsymbol{t}_i, \alpha), s_\wedge(\boldsymbol{t}_i, \alpha)\}$, where $s_/(\boldsymbol{t}_i, \alpha)$, $s_\setminus(\boldsymbol{t}_i, \alpha)$ are sampling functions with linearly increasing and decreasing time spans g_i, respectively, and $s_\vee(\boldsymbol{t}_i, \alpha) = [s_\setminus(\boldsymbol{t}_p, \alpha), s_/(\boldsymbol{t}_f, \alpha)]$, $s_\wedge(\boldsymbol{t}_i, \alpha) = [s_/(\boldsymbol{t}_p, \alpha), s_\setminus(\boldsymbol{t}_f, \alpha)]$ are combinations of $s_/$ and $s_\setminus$, where $\boldsymbol{t}_p$ and $\boldsymbol{t}_f$ represent the past and future segments of the time series, respectively, and $[\bullet, \bullet]$ is the concatenation operation.

To obtain linearly increasing or linearly decreasing time spans $\boldsymbol{g} = \{g_1, g_2, \ldots, g_l\}$, one possible solution would be to iterate through all possible time span combinations $\{\boldsymbol{g}_1, \boldsymbol{g}_2, \ldots, \boldsymbol{g}_n\}$ where $n = \binom{T-1}{l-1}$ under restriction $\sum_{i=1}^{l} g_i = T$, $g_i \in \mathcal{Z}^+$ and find a time span such that (1) obtains a maximum or minimum value, which is used to measure the linearity of $\boldsymbol{g}$

$$\rho(\boldsymbol{g}) = \frac{\sum_{j=1}^{l}(g_j - \bar{g})(j - \bar{j})}{\sqrt{\sum_{j=1}^{l}(g_j - \bar{g})^2}\sqrt{\sum_{j=1}^{l}(j - \bar{j})^2}}. \tag{1}$$

However, as the combination exploded, enumerating all possible time spans is quite time-consuming, where the time complexity is $\Theta((1+\epsilon)^{T-1})$ (ϵ is infinitesimal).

To address the abovementioned combination explosion problem, we define the generation of time span as an optimization problem by using (1) as objective function, $\sum_{i=1}^{l} g_i = T$, $g_i \in \mathcal{Z}^+$ as constraint. In this work, for $s_/(\boldsymbol{t}_i, \alpha)$ time sampling function, as shown in (2), we maximize (1)

$$\begin{aligned} &\max \quad \rho(\boldsymbol{g}) \\ &s.t. \sum_{i=1}^{l} g_i = T, g_i \in \mathcal{Z}^+. \end{aligned} \tag{2}$$

For $s_\setminus(\boldsymbol{t}_i, \alpha)$ sampling function, as shown in (3), we minimize (1)

$$\begin{aligned} &\min \quad \rho(\boldsymbol{g}) \\ &\text{s.t.} \sum_{i=1}^{l} g_i = T, g_i \in \mathcal{Z}^+. \end{aligned} \tag{3}$$

Algorithm 1: Irregular Time Sampling.

Require:
$\boldsymbol{t}_i$: A T-length time series, α: Sampling rate
$\boldsymbol{g}$: time spans
Ensure:
$\boldsymbol{t}'_i$: Transformation of $\boldsymbol{t}_i$ from irregular time sampling
function SAMPLING $(\boldsymbol{g})$ ▷ Generate sampling points
  $p_0 = 0$ ▷ Starting sampling point
  for each g_i in $\boldsymbol{g}$ **do**
    $p_i = p_{i-1} + g_i$ ▷ Calculation of sampling points
  end for
  return $(p_1, p_2, \ldots, p_l)$
end function
  function $s_{/}\boldsymbol{t}_i, \alpha$
  $l = \lceil T \cdot \alpha \rceil$
  $\boldsymbol{p} = \text{sampling}(\boldsymbol{g})$ ▷ $\boldsymbol{g}$ obtained by (2)
return $(t_{(i,p_1)}, t_{(i,p_2)}, \ldots t_{(i,p_l)})$
end function
function $s_{\backslash}\boldsymbol{t}_i, \alpha$
  $l = \lceil T \cdot \alpha \rceil$
  $\boldsymbol{p} = \text{SAMPLING}(\boldsymbol{g})$ ▷ $\boldsymbol{g}$ obtained by (3)
  return $(t_{(i,p_1)}, t_{(i,p_2)}, \ldots t_{(i,p_l)})$
end function
function $s_{\vee}\boldsymbol{t}_i, \alpha$
  $\boldsymbol{t}_{i,p} = (t_{(i,1)}, \ldots, t_{i,\lceil T/2 \rceil})$ ▷ Get past segment
  $\boldsymbol{t}_{i,f} = (t_{(i,\lceil T/2 \rceil+1)}, \ldots, t_{(i,T)})$ ▷ Get future segment
  return $[s_{\backslash}(\boldsymbol{t}_{i,p}, \alpha), s_{/}(\boldsymbol{t}_{i,f}, \alpha)]$
end function
function $s_{\wedge}\boldsymbol{t}_i, \alpha$
  $\boldsymbol{t}_{i,p} = (t_{(i,1)}, \ldots, t_{i,\lceil T/2 \rceil})$ ▷ Get past segment
  $\boldsymbol{t}_{i,f} = (t_{(i,\lceil T/2 \rceil+1)}, \ldots, t_{(i,T)})$ ▷ Get future segment
  return $[s_{/}(\boldsymbol{t}_{i,p}, \alpha), s_{\backslash}(\boldsymbol{t}_{i,f}, \alpha)]$
end function

Obviously, the number of solutions can be divided into two cases, 1) multiple solutions, and 2) unique solutions. For example, when $l = 2, T = 5$, all possible solutions will result in a maximum or minimum value of $\rho(\boldsymbol{g})$. If $l = 4, T = 5$ maximizing $\rho(\boldsymbol{g})$ or minimizing $\rho(\boldsymbol{g})$ both have only one solution, where $\rho(\boldsymbol{g})$ obtains the maximum 0.774597 when the solution $\boldsymbol{g} = \{1, 1, 1, 2\}$ and the minimum -0.774597 when $\boldsymbol{g} = \{2, 1, 1, 1\}$. For the multisolution case, we randomly choose one of the solutions. After obtaining the time span $\boldsymbol{g} = \{g_1, g_2, \ldots, g_l\}$ by maximizing or minimizing (1), the sampling points $\{p_1, p_2, \ldots, p_l\}$ are subsequently calculated by $p_j = p_{j-1} + g_j, p_0 = 0$, and finally the transformed time series $\boldsymbol{t}'_i = (t_{(i,p_1)}, t_{(i,p_2)}, \ldots, t_{(i,p_l)})$ is obtained.

B. Training on Labeled Data

Given labeled time series set $\mathcal{D}_L$, the time sampling function set $\mathcal{S}$ with α sampling rate, the backbone encoder f_θ, and the supervised classifier h_μ. We first sample the input time series $\boldsymbol{t}_i$ to get the sampled time series $\boldsymbol{t}'_{i,j} = s_j(\boldsymbol{t}_i, \alpha)$ and then fed the sampled time series $\boldsymbol{t}'_{i,j}$ to the backbone encoder f_θ to get feature representation $\mathbf{Z}_{i,j} = f_\theta(\boldsymbol{t}'_{i,j})$. Finally, the prediction $p_{i,j} = h_\mu(\mathbf{Z}_{i,j})$ is made based on the obtained feature $\mathbf{Z}_{i,j}$. The supervised training loss (4) is defined as a cross-entropy loss

$$L_{\text{sup}} = -\frac{1}{|\mathcal{D}_L| \cdot |\mathcal{S}|} \sum_{i=1}^{|\mathcal{D}_L|} \sum_{j=1}^{|\mathcal{S}|} y_i \cdot \log(p_{i,j}). \tag{4}$$

C. Training on Unlabeled Data

To explore unlabeled data in temporal structure learning, we use the self-generated time sampling function as supervised learning signals and conduct transformation prediction tasks on unlabeled sampled data. Given unlabeled time series datasets $\mathcal{D}_U$, set $\mathcal{S}$ of time sampling functions with α sampling rate. Each unlabeled time series $\boldsymbol{t}_i$ will be sampled by the time sampling function $s_j \in \mathcal{S}$ to obtain the sampled time series $\boldsymbol{t}'_{i,j}$. Input the sampled time series into the backbone encoder f_θ with shared weights to extract the feature embedding $\mathbf{Z}_{i,j} = f_\theta(\boldsymbol{t}'_{i,j})$, and then the obtained features are fed to the self-supervised transformation prediction head h_ϕ to get the final prediction $p_{i,j} = h_\phi(\mathbf{Z}_{i,j})$. Self-supervised training loss (5) is defined as follow:

$$L_{\text{usp}} = -\frac{1}{|\mathcal{D}_U| \cdot |\mathcal{S}|} \sum_{i=1}^{|\mathcal{D}_U|} \sum_{j=1}^{|\mathcal{S}|} j \cdot \log(p_{i,j}) \tag{5}$$

where j is the sampling function label. The model training algorithm is shown in Algorithm 2.

IV. Experiments

In this section, we describe the experimental settings, including datasets, baseline methods, and implementation. Then, we analysis the experimental results by comparing them with the state-of-the-art semisupervised methods.

A. Datasets

To verify the effectiveness of the proposed method, in the experiment, we use different categories of datasets, including four public datasets SmallKitchenAppliances, CricketX, Phoneme, and FordB from *UCR Time Series Archive*[1], as well as a real-world bearing dataset MFPT[2] and an EEG dataset EpilepticSeizure.[3] All selected datasets have a different number of instances, signal lengths, and classes. In the experiment, the proportion of the training set, validation set, and test set are set to 60%, 20%, and 20%, respectively. The details of the datasets are shown in Table I.

B. Baseline Methods

We compare the proposed iTimes against some baselines and several state-of-the-art semisupervised methods:

[1][Online]. Available: https://www.cs.ucr.edu/~eamonn/time_series_data_2018/
[2][Online]. Available: https://www.mfpt.org/fault-data-sets/
[3][Online]. Available: https://archive.ics.uci.edu/ml/datasets/Epileptic+Seizure+Recognition

Algorithm 2: Model Training.

Require:
Labeled data $(\boldsymbol{t}_i, y_i) \in \mathcal{D}_L$, and unlabeled input $\boldsymbol{t}_i \in \mathcal{D}_U$.
Backbone encoder f_θ; Classification head h_μ;
Transformation prediction head h_φ; Learning rate η.
Sampling rate α, Sampling function set $\mathcal{S}$.

for each epoch **do**
  for each labeled minibatch B_L **do**
    $\boldsymbol{t}'_{i,j} = s_j(\boldsymbol{t}_{i\in B_L}, \alpha)$ ▷ Irregularly sampling labeled data
    $\mathbf{Z}_{i,j} = f_\theta(\boldsymbol{t}'_{i,j})$ ▷ Embedding of labeled inputs.
    $p_{i,j} = h_\mu(\mathbf{Z}_{i,j})$ ▷ Label classification.
    $\mathcal{L}_{\text{sup}} = -\frac{1}{|B_L|\cdot|\mathcal{S}|}\sum_{i=1}^{|B_L|}\sum_{j=1}^{|\mathcal{S}|} y_i \cdot \log(p_{i,j})$ ▷ Cross-entropy loss.
    $\theta = \theta - \eta\nabla_\theta\mathcal{L}_{\text{sup}}, \mu = \mu - \eta\nabla_\mu\mathcal{L}_{\text{sup}}$ ▷ Update models.
  end for
  for each unlabeled minibatch B_U **do**
    $\boldsymbol{t}'_{i,j} = s_j(\boldsymbol{t}_{i\in B_U}, \alpha)$ ▷ Irregularly sampling unlabeled data
    $\mathbf{Z}_{i,j} = f_\theta(\boldsymbol{t}'_{i,j})$ ▷ Embedding of unlabeled inputs.
    $p_{i,j} = h_\varphi(\mathbf{Z}_{i,j})$ ▷ sampling function classification.
    $\mathcal{L}_{\text{usp}} = -\frac{1}{|B_L|\cdot|\mathcal{S}|}\sum_{i=1}^{|B_L|}\sum_{j=1}^{|\mathcal{S}|} j \cdot \log(p_{i,j})$ ▷ Cross-entropy loss.
    $\theta = \theta - \eta\nabla_\theta\mathcal{L}_{\text{usp}}, \varphi = \varphi - \eta\nabla_\varphi\mathcal{L}_{\text{usp}}$ ▷ Update models.
  end for
end for
return Backbone encoder f_θ and classification head h_μ

TABLE I
STATISTICS OF DATASETS

Dataset	# Sample	# Length	# Class	Category
SmallKitchenAppliances	750	720	3	Device
CricketX	780	300	12	Motion
Phoneme	2110	1024	39	Sensor
MFPT	2574	1024	15	Machine
FordB	3779	500	2	Sensor
EpilepticSeizure	11500	178	5	EEG

1) *Supervised:* A supervised learning baseline method using a ConvNet as encoder f_θ and a linear layer as classifier.
2) *Pseudolabel [13]:* is a SSL model that expands the supervised learning datasets by generating pseudolabels on unlabeled data.
3) Π-*Model [14]:* use self-ensembling to form a consensus prediction of the unknown labels using the outputs of the network-in-training on different epoch under different regularization and input augmentation conditions.
4) *MTL [8]:* is a SSL model that leverages features learned from the self-supervised task on unlabeled data.
5) *MixMatch [15]:* is a SSL model that integrates consistency regularization, entropy minimization, mixup in one.
6) *FixMatch [16]:* is a SSL model that leverages pseudolabel and consistency regularization learning features.
7) *SemiTime [9]:* is a SSL model, which learns the temporal relationship in time series by predicting the past–future relationship between past and future segments.

TABLE II
IMPLEMENTATION DETAIL OF ITIMES

	Layer description	Output Tensor Dim.
#0	Input time series (or time piece)	$1 \times T$ (or $1 \times L$)
	Backbone encoder	
#1	Conv1D(1, 8, 4, 2, 1)+BatchNorm+ReLU	$8 \times T/2$ (or $8 \times L/2$)
#2	Conv1D(8, 16, 4, 2, 1)+BatchNorm+ReLU	$16 \times T/4$ (or $16 \times L/4$)
#3	Conv1D(16, 32, 4, 2, 1)+BatchNorm+ReLU	$32 \times T/8$ (or $32 \times L/8$)
#4	Conv1D(32, 64, 4, 2, 1)+BatchNorm+ReLU +AvgPool1D+Flatten+Normalize	64
	Classification head	
#1	Linear+BatchNorm+LeakyReLU	256
#2	Linear+Sigmoid	C
	Prediction head	
#1	Linear+BatchNorm+LeakyReLU	256
#2	Linear+Softmax	4

C. Implementation

The programming environment of iTimes is based on Python 3.7 and PyTorch. All the experiments are conducted on Unbuntu 18.04 with Intel Core i7-10700 K, 64-GB RAM, and NVIDIA GeForce RTX 3080. ITimes uses the same backbone encoder f_θ as the benchmark models, a simple four-layer and one-dimensional neural network using the ReLu activation function and batch normalization. The supervised learning classifier h_μ and the self-supervised transformation prediction head h_ϕ use one linear layer with 256 neurons and two fully connected layers, respectively. The Adam optimizer with a learning rate of 0.01 is used as the training optimizer. All models were trained at 1000 iterations, and 128 batch sizes, and an early stopping callback strategy with 200 patient iterations was used to detect validation metrics and to end training when no boost was observed. We use time warping, and magnitude warping data augmentation [29] for all models. The effectiveness of the combination of irregular time sampling functions on the experimental results varies with the different datasets. We selected the best combination for each datasets through grid search and used classification accuracy as the evaluation metric. We start the experiments by computing the sampling points$\{p_1, p_2, \ldots, p_l\}$ for all irregular time sampling functions. The detailed architectural diagrams of iTimes are shown in Table II.

D. Experimental Results and Analysis

1) Ablation Study: In this section, we assess the effectiveness and stability of iTimes by investigating the impact of different combinations of time sampling functions, sampling rates, backbone encoders, and embedding vector sizes on time series classification.

We first examine the effect of the combination of two-time sampling functions on the classification accuracy of the time series with 10% labeled data under 80% sampling rate and 64 embedding size. As shown in Fig. 4, MFPT dataset and EpilepticSeizure dataset obtains the best classification accuracy

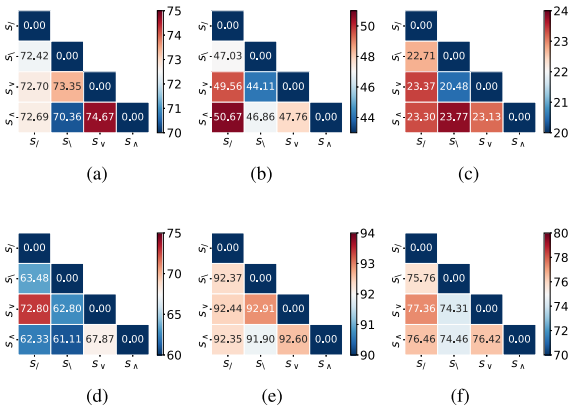

Fig. 4. Impact of different compositions of two time sampling functions with sampling rate (α) 80% on different datasets (10% labeled data). (a) SmallkitchenAppliances. (b) CricketX. (c) Phoneme. (d) MFPT. (e) FordB. (f) EpilepticSeizure.

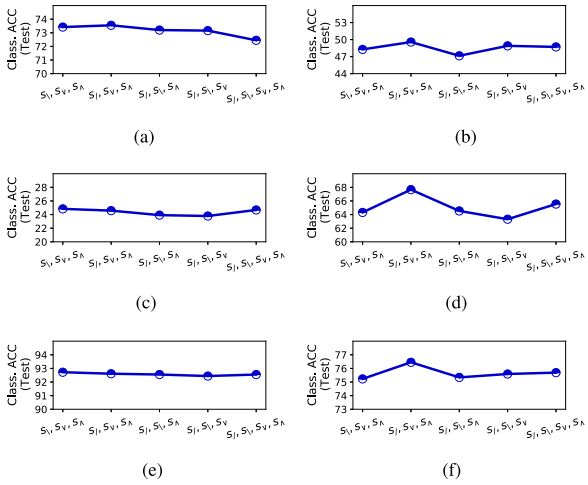

Fig. 5. Impact of different combinations of time sampling functions with sampling rate (α) 80% on different datasets (10% labeled data). (a) SmallkitchenAppliances. (b) CricketX. (c) Phoneme. (d) MFPT. (e) FordB. (f) EpilepticSeizure.

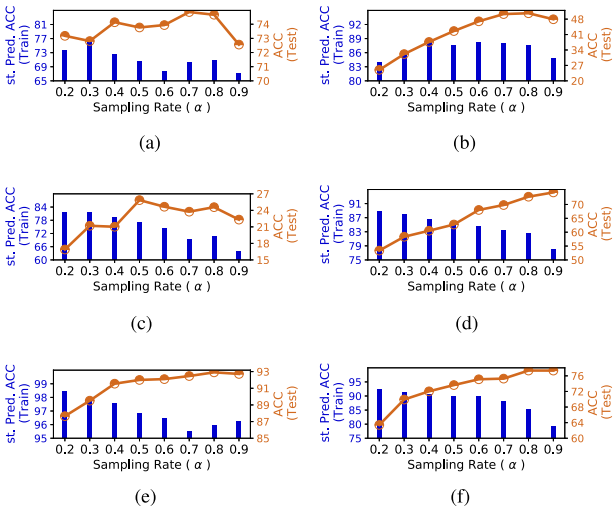

Fig. 6. Impact of different sampling rate(α) on different datasets (10% labeled data). (a) SmallkitchenAppliances. (b) CricketX. (c) Phoneme. (d) MFPT. (e) FordB. (f) EpilepticSeizure.

under the combination of ($s_/$, $s_\vee$), CricketX, FordB, and SmallKitchenAppliances dataset obtains the best classification under the combination of ($s_/$, $s_\wedge$), ($s_\backslash$, $s_\vee$), and ($s_\vee$, $s_\wedge$), respectively, and Phoneme dataset obtains the best classification accuracy under the combination of ($s_\backslash$, $s_\wedge$) time sampling function. It is evident that stable classification accuracies are achieved on the MFPT and SmallKitchenAppliances datasets for all combinations of two-time sampling functions, more stable classification accuracies are achieved on the two datasets containing the largest number of samples for FordB and EpilepticSeizure, and some fluctuations in classification accuracies are observed on the CricketX and Phoneme datasets. One possible reason for the fluctuations in classification accuracy in the two datasets is that the information distribution of the time series is not suitable for the time sampling function. For example, the time series may carry less information in the front than in the tail, and $s_/$ that focuses on more information in the front may not capture enough of the original information compared to a uniform distribution of information in the time series.

In addition to examining the combination of two-time sampling functions, we also examined the effect of the combination of three time sampling functions and all time sampling functions on time series classification. As shown in Fig. 5, compared with the combination of two-time sampling functions, the experimental results of three and all-time sampling function combinations do not have much variation, which demonstrates the stability of iTimes.

We next evaluate the effectiveness of the proposed irregular time sampling module by investigating the effect of different sampling rates α on time series classification with 10% labeled data, and 64 embedding sizes under the optimal combination of time sampling functions. As shown in Fig. 6, where the blue bar indicates transformation prediction accuracy on training data (St. Pred. ACC), and the line indicates classification accuracy on test data (ACC). As the sampling rate α increases, the classification accuracy keeps increasing, and after α increasing to a certain value, the classification accuracy of all datasets except the MFPT dataset starts to decrease. We speculate that as the sampling rate increases, it will make each sampled time series carry more original information, which is beneficial for the model to learn the temporal structure. As the sampling rate continues to increase, the distinguishability of time series sampled by different time sampling functions becomes the less noticeable to the extent that self-supervised learning cannot learn the temporal structure.

We then evaluate the impact of different encoders, a simple four-layer CNN and two residual networks, on iTimes via within 10% labeled CricketX dataset. The experimental results are shown Table III, ResNet18, ResNet34 improved the classification accuracy by 6.09% and 10.90%, respectively, compared with SimConv4. Although ResNet obtained better experimental results, ResNet took up more GPU memory than SimConv4.

We finally evaluate the stability of iTimes by examining different backbone encoder embedding vector sizes $|\mathbf{Z}_{i,j}|$ with

TABLE III
IMPACT OF DIFFERENT ENCODERS ON ITIMES

–	SimConv4	ResNet18	ResNet34
CricketX	49.78±3.42	52.81±2.91	55.21±3.05

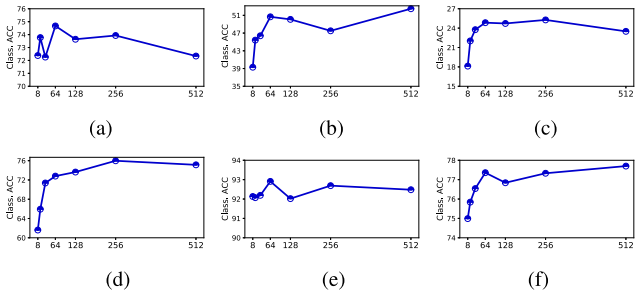

Fig. 7. Impact of different embedding size on different datasets (10% labeled data). (a) SmallkitchenAppliances. (b) CricketX. (c) Phoneme. (d) MFPT. (e) FordB. (f) EpilepticSeizure.

10% labeled data under 80% sampling rate. As shown in Fig. 7, where the line indicates classification accuracy on test data (Class. ACC). As the embedding size $|\mathbf{Z}_{i,j}|$ increases, the classification accuracy first increases and then stabilizes within a small fluctuation. The time series classification accuracy is poor at low-dimensional embedding because the embedding vector size is so tiny that the representation learned by iTimes is insufficient to express the semantic information of the original samples. The classification accuracy of iTimes achieves a steady state when the embedding vector size exceeds 64, reflecting the stability of iTimes. We also discovered that the EpilepticSeizure dataset with more than 10 000 samples achieved the maximum classification accuracy at 64 dimensions of embedding vector size, while the MFPT data with around 2000 samples achieved the maximum at 256 dimensions of embedding vector size, indicating that the embedding vector size is not correlated with the sample size. We conjecture that although the sample quantity of MFPT is tiny, each sample carries more diverse information compared to EpilepticSeizure, resulting in the 64-dimensional embedding vector size being insufficient to express the semantic information of the sample within MFPT.

2) Applications: In this section, we examine the performance of iTimes in practical applications on the MFPT and EpilepticSeizure datasets. MFPT is a bearing fault dataset that comprises one data from a bearing test rig and three real-world fault data, divided into 14 abnormal classes under different loads and one normal class. And EpilepticSeizure is a commonly used EEG data for seizure detection, it includes 11 500 instances with five classes (Eyes open, Eyes closed, Tumor, Healthy, and Seizure). The experimental results are shown in Table IV. The classification accuracy of iTimes exceeds that of all baseline methods. Given 10% labeled data, iTimes improves 36.43% and 13.28% over the Supervised method on the MFPT and EpilepticSeizure, respectively, and 13.71% and 11.29% over Fixmatch, respectively.

3) Time Series Classification: In this section, we evaluate iTimes by comparing it with other semisupervised-based

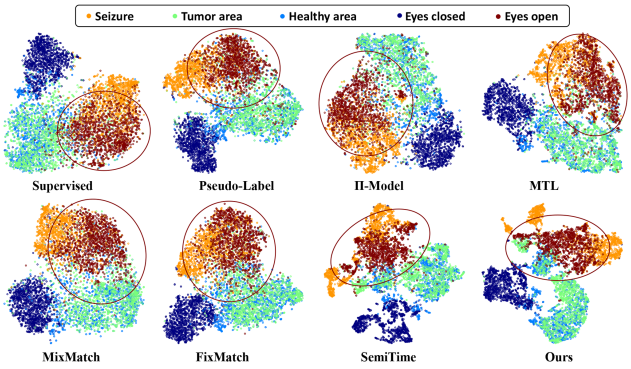

Fig. 8. t-SNE visualization of the temporal structure on EpilepticSeizure.

state-of-the-art models on time series classification tasks. Following the previous research [8], [14], we randomly select 10%, 20%, 40%, and 100% of the training set as labeled data and use the entire training set as unlabeled data for model training. As shown in Table IV, our proposed method consistently outperforms all benchmark models on all datasets. For example, given 10% labeled data, the accuracy is 4.90% higher than that of SemiTime and 6.68% higher than that of MixMatch on CricketX, 7.86% higher than that of FixMatch, and 8.78% higher than that of MTL on EpilepticSeizure, 7.89% higher than that of Π-Model on EpilepticSeizure. Given 20% labeled data, our method improves the accuracy over SemiTime by 4.26% on Phoneme, over Π-Model by 14.10% on SmallKitchenAppliances, and 2.39% on FordB, respectively. The results show that predicting unlabeled data by using self-ensembling or generating pseudolabel alone does not capture the potential temporal structure of time series effectively, which is of great significance to the semisupervised representation learning of time series, coarse-grained predicting whether past, future time series segments belong to the same time series, cannot fully explore the temporal relationship within time series. It can also be seen that the classification accuracy decreases slightly as the amount of labeled data decreases. But all exceed the supervised and other baselines, demonstrating the stability of the model. Moreover, we also use 100% training set as labeled and unlabeled data to evaluate the performance of our proposed method for supervised and self-supervised training, and the experimental results demonstrate that our method is consistently better than supervised learning and self-supervised learning based on MTL. This shows that the forecasting pretext task of MTL cannot capture the internal temporal structure of unlabeled time series effectively, while we designed transformation prediction is able to learn the internal structure of time series.

4) Visualization: To qualitatively analyze the learned temporal structure, we extract the representation embedding vector of the time series using a trained backbone encoder and visualize it in 2-D space using the t-SNE [30] in order to verify the semantic consistency of the learned embeddings. Fig. 8 shows the extracted representation embedding vectors by iTimes as well as the baseline models on the EpilepticSeizure dataset. As can be seen from Fig. 8, class Eyes open bounded by

TABLE IV
TEST CLASSIFICATION ACCURACY (%, AVERAGES OF FIVE SEEDS AND FIVE RUNS) FOR SUPERVISED BASELINE AND SSL ON DIFFERENT DATASETS

Label ratio	**10%**	**20%**	**40%**	**100%**	**10%**	**20%**	**40%**	**100%**	**10%**	**20%**	**40%**	**100%**
Dataset	SmallKitchenAppliances				CricketX				Phoneme			
Supervised	63.11±3.50	63.11±2.30	68.23±3.51	73.03±2.65	33.62±0.95	38.97±2.08	52.64±2.53	62.98±2.01	19.54±0.69	22.45±1.14	27.82±1.62	35.90±1.4
Pseudo-Label [13]	52.46±4.09	62.29±2.96	73.04±3.75	-	38.87±2.26	44.44±2.91	53.39±2.18	-	18.92±0.72	21.40±0.99	26.34±0.68	-
Π-Model [14]	57.04±1.86	60.93±2.82	65.44±1.85	-	38.61±2.29	48.18±2.07	54.73±1.04	-	20.71±0.65	22.21±1.71	25.74±1.78	-
MTL [8]	65.03±2.71	66.69±3.03	71.23±2.19	75.30±3.24	40.94±1.97	50.12±1.22	55.10±1.12	63.58±1.72	19.98±1.67	20.00±0.66	25.58±1.17	32.34±0.78
MixMatch [15]	55.57±2.58	55.55±1.59	59.76±1.35	-	43.10±2.34	56.80±3.16	64.46±1.66	-	23.09±0.63	26.89±1.02	31.53±0.40	-
FixMatch [16]	65.14±3.36	65.60±4.23	74.10±2.48	-	38.33±2.07	50.89±2.04	60.66±1.27	-	18.89±1.80	21.58±1.57	27.72±1.0	-
SemiTime [9]	69.35±2.08	70.55±2.06	71.57±2.10	74.58±2.67	44.88±3.13	51.61±0.66	58.71±2.78	65.66±1.58	20.64±1.34	25.28±1.42	30.11±1.19	36.96±1.63
iTimes	**74.13±4.91**	**75.03±4.94**	**77.38±2.41**	**81.23±1.28**	**49.78±3.42**	**58.64±2.59**	**68.37±2.75**	**77.71±1.40**	**24.23±1.53**	**29.54±2.12**	**36.54±0.42**	**45.12±1.2**
Dataset	MFPT				EpilepticSeizure				FordB			
Label ratio	**10%**	**20%**	**40%**	**100%**	**10%**	**20%**	**40%**	**100%**	**10%**	**20%**	**40%**	**100%**
Supervised	50.88±0.32	57.14±0.54	69.76±0.48	81.63±0.15	68.40±0.43	70.77±0.70	73.49±0.60	77.77±1.13	89.69±0.65	90.48±1.08	91.80±0.47	92.60±0.90
Pseudo-Label [13]	63.90±2.62	65.39±1.70	69.60±2.27	-	68.57±0.50	72.92±0.48	74.60±0.65	-	89.02±0.77	90.58±0.34	91.71±0.37	-
Π-Model [14]	55.41±0.65	59.68±0.43	70.15±1.01	-	69.60±0.34	71.58±0.64	74.54±0.55	-	88.94±0.74	90.64±0.37	91.02±0.14	-
MTL [8]	40.84±1.97	50.12±1.22	55.10±1.12	63.58±1.72	68.71±0.94	73.17±0.81	74.77±0.75	78.53±0.62	90.43±0.88	91.21±1.10	91.84±0.77	92.55±0.96
MixMatch [15]	52.56±1.01	68.62±0.80	81.87±1.60	-	67.99±0.09	69.67±0.30	72.22±0.63	-	90.75±0.44	91.34±0.39	91.72±0.58	-
FixMatch [16]	61.05±1.65	71.72±1.76	81.04±0.8	-	69.63±0.67	71.37±0.24	72.30±0.76	-	90.31±0.48	90.54±0.19	91.76±0.20	-
SemiTime [9]	64.16±0.85	69.84±0.94	76.49±0.54	84.33±0.50	74.86±0.42	75.54±0.63	77.01±0.79	79.26±1.02	90.97±0.88	91.46±0.68	91.80±0.48	92.40±0.74
iTimes	**69.42±1.23**	**78.84±1.87**	**85.78±1.45**	**92.03±0.45**	**77.49 ± 0.49**	**78.93 ± 0.54**	**82.23 ± 0.56**	**82.87 ± 0.56**	**92.08±1.06**	**93.03±0.58**	**93.35±0.57**	**94.29±0.47**

Notes: All methods use the same four-layer convolutional backbone.

circle obtains a better clustering performance compared to the other baselines. Thus by fine-grained examining the temporal relationships within unlabeled time series, iTimes is able to learn more useful embeddings and enables better clustering ability of time series, maintaining more semantic consistency. Interestingly, we also found that EEG recordings from healthy brain regions and tumor brain regions were not clustered very well, despite the better performance of our proposed model, but this finding provides additional insights for data inspection and model optimization.

V. CONCLUSION

In this article, we propose a semisupervised time series classification framework to explore the underlying temporal structure within unlabeled time series in a self-supervised manner. We design a simple but efficient irregular time sampling strategy and extract the underlying temporal structure features inside the time series by predicting the transformation type of irregular time sampling. Our main finding is that the examination of fine-grained temporal relationships of unlabeled time series is more conducive to the utilization of unlabeled time series for SSL. While conducting this study, we expected to find a perfect combination of time sampling functions to examine the temporal structure. For the sake of simplicity of implementation, we only selected $s_{/}, s_{\backslash}, s_{\vee}, s_{\wedge}$ from the infinite time sampling functions, which will undoubtedly limit the feature learning capability of the ConvNets. We intend to explore more sampling transformation methods in the future to find the most efficient transformation.

REFERENCES

[1] Y. Peng, F. Jin, W. Kong, F. Nie, B.-L. Lu, and A. Cichocki, "Ogssl: A semi-supervised classification model coupled with optimal graph learning for EEG emotion recognition," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 30, pp. 1288–1297, May 2022, doi: 10.1109/TNSRE.2022.3175464.

[2] R. Rodrigues and P. Couto, "Semi-supervised learning for ECG classification," in *Proc. Comput. Cardiol.*, vol. 48, pp. 1–4, 2021, doi: 10.23919/CinC53138.2021.9662693.

[3] Q. Zhu, Z. Chen, and Y. C. Soh, "A novel semisupervised deep learning method for human activity recognition," *IEEE Trans. Ind. Informat.*, vol. 15, no. 7, pp. 3821–3830, Jul. 2019, doi: 10.1109/TII.2018.2889315.

[4] X. Zhao, M. Jia, and Z. Liu, "Semisupervised graph convolution deep belief network for fault diagnosis of electormechanical system with limited labeled data," *IEEE Trans. Ind. Informat.*, vol. 17, no. 8, pp. 5450–5460, Aug. 2021, doi: 10.1109/TII.2020.3034189.

[5] Y. Yang, J. Zhong, W. Li, T. A. Gulliver, and S. Li, "Semisupervised multilabel deep learning based nonintrusive load monitoring in smart grids," *IEEE Trans. Ind. Informat.*, vol. 16, no. 11, pp. 6892–6902, Nov. 2020, doi: 10.1109/TII.2019.2955470.

[6] T. Ko and H. Kim, "Fault classification in high-dimensional complex processes using semi-supervised deep convolutional generative models," *IEEE Trans. Ind. Informat.*, vol. 16, no. 4, pp. 2868–2877, Apr. 2020, doi 10.1109/TII.2019.2941486.

[7] J. Yi and J. Park, "Semi-supervised bearing fault diagnosis with adversarially-trained phase-consistent network," in *Proc. 27th ACM SIGKDD Conf. Knowl. Discov. Data Mining*, 2021, pp. 3875–3885.

[8] S. Jawed, J. Grabocka, and L. Schmidt-Thieme, "Self-supervised learning for semi-supervised time series classification," in *Pacific-Asia Conf. Knowl. Discov. Data Mining*, 2020, pp. 499–511.

[9] H. Fan, F. Zhang, R. Wang, X. Huang, and Z. Li, "Semi-supervised time series classification by temporal relation prediction," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2021, pp. 3545–3549.

[10] Y. Chen, B. Hu, E. Keogh, and G. E. Batista, "DTW-D: Time series semi-supervised learning from a single example," in *Proc. 19th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2013, pp. 383–391.

[11] L. de Carvalho Pagliosa and R. F. de Mello, "Semi-supervised time series classification on positive and unlabeled problems using cross-recurrence quantification analysis," *Pattern Recognit.*, vol. 80, pp. 53–63, 2018.

[12] Y. Grandvalet and Y. Bengio, "Semi-supervised learning by entropy minimization," in *Proc. Adv. Neural Inf. Process. Syst.*, 2005, pp. 529–536.

[13] D.-H. Lee et al., "Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks," in *Proc. Workshop Challenges Representation Learn.*, 2013, vol. 3, no. 2.

[14] S. Laine and T. Aila, "Temporal ensembling for semi-supervised learning," in *Proc. 5th Int. Conf. Learn. Representations*, 2017.

[15] D. Berthelot, N. Carlini, I. Goodfellow, N. Papernot, A. Oliver, and C. Raffel, "Mixmatch: A holistic approach to semi-supervised learning," 2019, *arXiv:1905.02249*.

[16] K. Sohn et al., "Fixmatch: Simplifying semi-supervised learning with consistency and confidence," 2020, *arXiv:2001.07685*.

[17] R. Sakurai, S. Yamane, and J.-H. Lee, "Restoring aspect ratio distortion of natural images with convolutional neural network," *IEEE Trans. Ind. Informat.*, vol. 15, no. 1, pp. 563–571, Jan. 2019, doi: 10.1109/TII.2018.2803041.

[18] S. Gidaris, P. Singh, and N. Komodakis, "Unsupervised representation learning by predicting image rotations," in *Proc. 6th Int. Conf. Learn. Representations*, 2018.

[19] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *Proc. 37th Int. Conf. Mach. Learn.*, 2020.

[20] D. Wei, J. J. Lim, A. Zisserman, and W. T. Freeman, "Learning and using the arrow of time," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pp. 8052–8060, 2018.

[21] Y. Yao, C. Liu, D. Luo, Y. Zhou, and Q. Ye, "Video playback rate perception for self-supervised spatio-temporal representation learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020.

[22] J. Wang, J. Jiao, and Y.-H. Liu, "Self-supervised video representation learning by pace prediction," in *Proc. Euro. Conf. Comput. Vis.*, 2020.

[23] S. Pascual, M. Ravanelli, J. Serrá, A. Bonafonte, and Y. Bengio, "Learning problem-agnostic speech representations from multiple self-supervised tasks," in *Proc. Conf. Int. Speech Commun. Assoc.*, 2019, pp. 161–165.

[24] A. v. d. Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," 2018, *arXiv:1807.03748*.
[25] H. Fan, F. Zhang, and Y. Gao, "Self-supervised time series representation learning by inter-intra relational reasoning," 2020, *arXiv:2011.13548*.
[26] A. Saeed, F. D. Salim, T. Ozcelebi, and J. Lukkien, "Federated self-supervised learning of multisensor representations for embedded intelligence," *IEEE Internet Things J.*, vol. 8, no. 2, pp. 1030–1040, Jan. 2021, doi: 10.1109/JIOT.2020.3009358.
[27] D. Spathis, I. Perez-Pozuelo, S. Brage, N. J. Wareham, and C. Mascolo, "Self-supervised transfer learning of physiological representations from free-living wearable data," in *Proc. Conf. Health, Inference, Learn.*, 2021, pp. 69–78.
[28] P. Sarkar and A. Etemad, "Self-supervised ecg representation learning for emotion recognition," *IEEE Trans. Affect. Comput.*, early access, Aug. 6, 2020, doi: 10.1109/TAFFC.2020.3014842.
[29] T. T. Um et al., "Data augmentation of wearable sensor data for parkinsons disease monitoring using convolutional neural networks," in *Proc. 19th ACM Int. Conf. Multimodal Interaction*, 2017, pp. 216–220.
[30] P. G. Poličar, M. Stražar, and B. Zupan, "Opentsne: A modular python library for T-SNE dimensionality reduction and embedding," *BioRxiv*, 2019, Art. no. 731877.

Xuxin Liu received the B.E. degree in packaging engineeering from the Harbin University of Commerce, Harbin, China, in 2020. He is currently working toward the M.S. degree in computer science and technology with the Harbin University of Science and Technology, Harbin, China.

His research interests include time series analysis, network security, and machine learning.

Fengbin Zhang received the Ph.D. degree in computer application from Harbin Engineering University, Harbin, China, in 2005.

He is currently a Supervisor and Professor with the Harbin University of Science and Technology, Harbin, China. His research interests include network and information security, firewall technology, and intrusion detection technology.

Han Liu received the M.S. degree in computer science and technology from the Harbin University of Science and Technology, Harbin, China, in 2021. He is currently a Ph.D. candidate with Harbin University of Science and Technology.

His research interests include artificial intelligence, network security, and machine learning.

Haoyi Fan received the Ph.D. degree from the School of Computer Science and Technology, Harbin University of Science and Technology, Harbin, China, in 2021.

He is an Associate Research Fellow with the School of Computer and Artificial Intelligence, Zhengzhou University, Zhengzhou, China. His research interests include time series analysis, data mining, and deep learning.