Self-supervised Contrastive Representation Learning for Semi-supervised Time-Series Classification

Emadeldeen Eldele, Mohamed Ragab, Zhenghua Chen, Min Wu, Chee-Keong Kwoh, Xiaoli Li and Cuntai Guan *Fellow, IEEE*

Abstract—Learning time-series representations when only unlabeled data or few labeled samples are available can be a challenging task. Recently, contrastive self-supervised learning has shown great improvement in extracting useful representations from unlabeled data via contrasting different augmented views of data. In this work, we propose a novel **T**ime-**S**eries representation learning framework via **T**emporal and **C**ontextual **C**ontrasting (**TS-TCC**) that learns representations from unlabeled data with contrastive learning. Specifically, we propose time-series specific weak and strong augmentations and use their views to learn robust temporal relations in the proposed temporal contrasting module, besides learning discriminative representations by our proposed contextual contrasting module. Additionally, we conduct a systematic study of time-series data augmentation selection, which is a key part of contrastive learning. We also extend TS-TCC to the semi-supervised learning settings and propose a **C**lass-**A**ware TS-**TCC** (**CA-TCC**) that benefits from the available few labeled data to further improve representations learned by TS-TCC. Specifically, we leverage robust pseudo labels produced by TS-TCC to realize class-aware contrastive loss. Extensive experiments show that the linear evaluation of the features learned by our proposed framework performs comparably with the fully supervised training. Additionally, our framework shows high efficiency in few labeled data and transfer learning scenarios. The code is publicly available at https://github.com/emadeldeen24/TS-TCC.

Index Terms—self-supervised learning, semi-supervised learning, time-series classification, temporal contrasting, contextual contrasting, augmentation.

1 Introduction

Time-series data are being incrementally collected on daily basis from IoT, wearable devices and machines sensors for various applications in healthcare, manufacturing, etc. [1]. However, unlike images, time-series data generally do not contain human recognizable patterns to differentiate different classes and thus easily assign class labels to data. Consequently, different time-series applications require trained specialists to perform the challenging data annotation/labeling. Therefore, little time-series data have been labeled in real-world applications compared to the collected data [2]. With the advance of deep learning techniques, it becomes essential to train deep learning-based models with a massive amount of labeled data to learn useful representations and avoid overfitting. However, applying them to time-series data becomes very challenging with the aforementioned labeling limitations.

Self-supervised learning has gained much attention recently to make use of unlabeled data and learn powerful representations for deep learning models. Compared with models trained on fully labeled data (i.e., supervised models), self-supervised pretrained models can extract effective representations and achieve comparable performance when fine-tuned with a small percentage of the full labels [3], [4]. The recent rebirth in self-supervised learning relied on manually-designed pretext tasks to learn representations about data. For example, Noroozi et al. split the image into smaller patches, shuffled them, and trained the model to reorder these patches to form the original image [3]. Some other pretext tasks assigned pseudo labels to different vari-

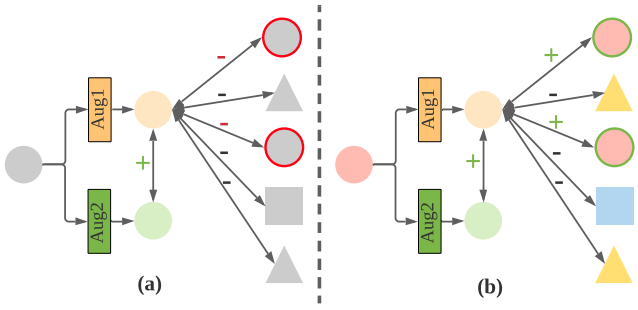


Fig. 1: (a) Unsupervised contrasting vs (b) supervised contrasting. In usual unsupervised contrasting, all the samples in the mini-batch are considered as negative pairs even if they belong to the same class. This could be avoided in the supervised contrasting.

- *Emadeldeen Eldele, Mohamed Ragab, Chee-Keong Kwoh, and Cuntai Guan are with the School of Computer Science and Engineering, Nanyang Technological University, Singapore (Email: {emad0002, asckkwoh, ctguan}@ntu.edu.sg, mohamedr002@e.ntu.edu.sg)*
- *Zhenghua Chen, Min Wu and Xiaoli Li are with the Institute for Infocomm Research, A*STAR, Singapore, 138632 (e-mail: chen0832@e.ntu.edu.sg; {wumin, xlli}@i2r.a-star.edu.sg).*

- *Both first and second authors are supported by A*STAR SINGA Scholarship. Min Wu is the corresponding author.*
- *This research is supported by the Agency for Science, Technology and Research (A*STAR) under its AME Programmatic Funds (Grant No. A20H6b0151) and Career Development Award (Grant No. C210112046).*

ations of input samples. For instance, Gidaris et al. applied several rotations to the original image and assigned a label to each angle [5]. The model was then trained to predict the rotation angle of the image as a classification task.

Contrastive learning has recently shown its strong ability over pretext tasks to learn representations from unlabeled data. The strength point of contrastive learning is its ability to learn invariant representations by contrasting different views of the input sample, which are generated using augmentation techniques [4], [6], [7]. However, these image-based contrastive learning methods may not able to work well on time-series data for the following reasons. First, unlike images, where its features are mostly spatial, we find time-series data are mainly characterised by the temporal dependencies [8]. Therefore, applying the aforementioned techniques directly to time-series data may not efficiently address the temporal features of data. Second, some augmentation techniques used for images such as color distortion, generally cannot fit well with time-series data. So far, few works on contrastive learning have been proposed for time-series data. For example, [9], [10] developed contrastive learning methods for bio-signals. However, these two methods are proposed for specific clinical applications and may not generalize to other time-series data.

In this paper, we propose a novel framework that incorporates contrastive learning into self- and semi-supervised learning. Specifically, we propose a **T**ime-**S**eries representation learning framework via **T**emporal and **C**ontextual **C**ontrasting (TS-TCC) that is trained on totally unlabeled datasets. Our TS-TCC employs two contrastive learning and augmentation techniques to handle the temporal dependencies of time-series data. We propose simple yet efficient data augmentations that can fit any time-series data to create two different, but correlated views of the input samples. These views are then used by the two innovative contrastive learning modules. In the first module, we propose a novel temporal contrasting module to learn *robust* representations by designing a tough cross-view prediction task. Specifically, for a certain timestep, it utilizes the past latent features of one augmentation to predict the future of the other augmentation. This novel operation will force the model to learn robust representation by a harder prediction task against any perturbations introduced by different timesteps and augmentations. In the second module, we propose contextual contrasting to further learn *discriminative* representations upon the robust representations learned by the temporal contrasting module. In this contextual contrasting module, we aim to maximize the similarity among different contexts of the same sample while minimizing similarity among contexts of different samples. The pretrained model learns powerful representations about the time-series data regardless of downstream tasks.

One limitation of the contextual contrasting is that the contrasting samples from the same class will be treated as negative pairs since label information is not available, which may deteriorate the performance. Therefore, we propose another variant for **C**lass-**A**ware TS-TCC (**CA-TCC**) to utilize class information when contrasting between samples. In particular, we rely on the generated pseudo labels by our pretrained TS-TCC. These pseudo labels are leveraged in a supervised contextual contrasting module to maximize the similarity between samples from the same classes together, while pushing samples apart if they are from different classes.

This journal paper is an extended version of our previous work [11]. The main extensions are listed as follows: 1) We extend our model to fit the semi-supervised setting, as we take advantage of the available few labeled data to further improve the learned representations. This is important as in many real-world applications, in which we could have limited labeled training examples. It is thus meaningful to further push the performance boundary by leveraging them. 2) We study time-series data augmentation, which is a key part of contrastive learning in more details and provide a systematic methodology of selecting the best augmentations for time-series data. 3) We conduct extensive experiments to assess CA-TCC, demonstrating the effectiveness of the learned representations in improving the different real-world downstream tasks. In summary, the main contributions of this work are as follows.

- A novel contrastive learning framework for time-series representation learning with two variants is developed. The first (TS-TCC) is proposed for learning from unlabeled data, and the second (CA-TCC) is developed to learn improved representations in the semi-supervised settings.
- We provide simple yet efficient augmentations that are designed for time-series data in the contrastive learning framework, and provide extensive discussion about their selection.
- We propose a novel temporal contrasting module to learn robust representations from time-series data by designing a tough cross-view prediction task. In addition, we propose a (supervised) contextual contrasting module to further learn discriminative representations upon the robust representations.
- We perform extensive experiments on the two proposed variants of our framework using three real-world datasets. Experimental results show that both variants are capable of learning effective representations for different scenarios.

2 Related Works

2.1 Self-supervised Learning

Self-supervised learning is gaining more attention recently, as it deals with the emerged problem of having few labeled data [12], [13], [14]. The recent advances in self-supervised learning started with applying pretext tasks on images to learn useful representations. Many pretext tasks, that vary in nature, have been proposed according to the target applications. For image related applications, Zhang et al. trained their model to color the input grayscale image [15]. Misra et al. proposed learning invariant representations based on pretext tasks by encouraging the representation of both the pretext task and the original image to be similar [16]. Zhai et al. developed pretext tasks based self-supervised training to empower the representations learned in a semi-supervised setting [17].

For video related tasks, Srivastava et al. proposed different tasks such as input sequence reconstruction, or future sequence prediction to learn representations of video

sequences [18]. In addition, Wei et al. proposed training the model to check if the order of the input frame sequences is correct or not [19]. Another direction is to use ranking as a proxy task to solve regression problems [20], in which the authors developed a backpropagation technique for Siamese networks to prevent the redundant computations. Although using pretext tasks can improve the representation learning, they are found to limit the generality of the learned representations. For example, classifying the different rotation angles of an image may deviate the model from learning features about the color or orientation of objects [21].

On the other hand, contrastive methods intend to learn invariant representations from different augmented views of data. The different methods vary between each other in their ways of choosing negative samples against positive samples during training. SimCLR [4] considered the augmented views of the original image as the positives, while all the other views of different images within the batch are treated as negatives. This technique benefits from larger batch sizes to accumulate more negative samples. Another approach was to accumulate a large number of the negative samples in a memory bank as proposed in MoCo [7]. The embeddings of these samples are updated with the most recent ones at regular intervals. The next direction was to use contrastive learning without the need of negative samples. For example, BYOL [22] learned representations by bootstrapping representations from two neural networks that interact and learn from each other. SimSiam [23] supported the idea of neglecting the negative samples, and relied only on a Siamese network and stop-gradient operation to achieve the state-of-the-art performance.

Some methods deployed contrastive learning in the semi-supervised settings. For example, FixMatch [24] used weak augmented views to produce pseudo labels for the input image, and if the pseudo label exceeds a confidence threshold, they use it to penalize the prediction of strong augmented view of the same input image. While all these approaches have successfully improved representation learning for visual data, they may not work well on time-series data that have different properties, such as temporal dependency.

2.2 Self-supervised Learning for Time-Series

Self-supervised representation learning for time-series is becoming more popular recently. Some approaches employed pretext tasks for time-series data. For example, Saeed et al. designed a binary classification pretext task for human activity recognition by applying several transformations on the data, and trained the model to classify between the original and the transformed versions [25]. Similarly, Sarkar at al. proposed SSL-ECG, in which ECG representations are learned by applying six transformations to the dataset as pretext tasks, and assigned pseudo labels according to the transformation type [26]. The model is expected to learn useful representations about the data by classifying these transformations. The same approach was used by Saeed et al. as they designed eight auxiliary tasks and learning representations from multi-sensor human activity data [27]. Additionally, Aggarwal et al. learned subject-invariant representations by modeling local and global activity patterns [28].

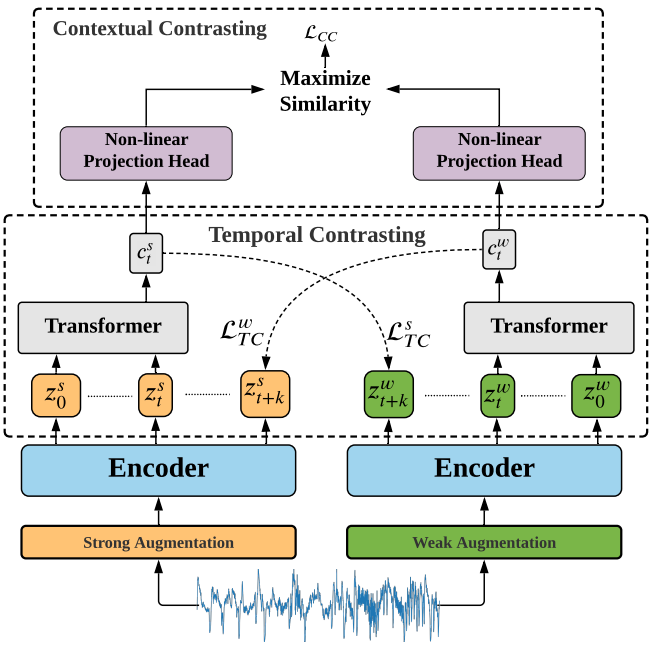


Fig. 2: Overall architecture of the proposed TS-TCC. The Temporal Contrasting module learns robust temporal features by a tough cross-view prediction task. The Contextual Contrasting learns discriminative features by maximizing the similarity between the contexts of the same sample, while minimizing its similarity with the other samples within the mini-batch.

Inspired by the success of contrastive learning in visual applications, few works have recently leveraged contrastive learning for time-series data. For example, CPC [21] learned representations by predicting the future in the latent space and showed great advances in various speech recognition tasks. Banville et al. studied three self-supervised tasks with two pretext tasks, i.e., relative positioning, temporal shuffling, and one contrastive task that uses CPC to learn representations about clinical EEG data [29]. They found that representations learned by CPC perform the best. This indicates that contrastive learning techniques generally perform better than the pretext tasks. Mohsenvand et al. designed EEG related augmentations and extended SimCLR model [4] to different EEG clinical data [9].

Existing approaches used either temporal or global features. Differently, we address both types of features in our cross-view temporal and contextual contrasting modules. These modules rely on different views for input data that we provide via time-series specific augmentations.

3 Methods

This section describes the components of the proposed framework in details. Starting with TS-TCC, we first generate two different yet correlated views of the input data based on strong and weak augmentations. Then, a temporal contrasting module is proposed to explore the temporal features of the data with two autoregressive models. These models perform a tough cross-view prediction task by predicting the future of one view using the past of the other. We further maximize the agreement between the contexts of the autoregressive models by a contextual contrasting module.

These components are illustrated in Fig. 2. For the semi-supervised settings, our proposed CA-TCC leverages supervised contextual contrasting module, as shown in Fig. 4. In the next subsections, we will introduce each component in more details.

3.1 Time-Series Data Augmentation

Data augmentation is a key part in the success of the contrastive learning methods [4], [22]. Contrastive methods try to maximize the similarity among different views of the same sample, while minimizing its similarity with other samples. It is thus important to design proper data augmentations for contrastive learning. Usually, contrastive learning methods use two (random) variants of the same augmentation [4], [9]. Specifically, given a sample x, they produce two views x_1 and x_2 sampled from the same augmentation family $\mathcal{T}$, i.e., $x_1 \sim \mathcal{T}$ and $x_2 \sim \mathcal{T}$. However, we argue that producing views from different augmentations can improve the robustness of the learned representations. Consequently, we propose two separate augmentations, such that one augmentation is weak and the other is strong.

Our framework uses strong augmentation to enable the *tough* cross-view prediction task in the next module, which helps in learning robust representations about the data. The weak augmentation aims to add some small variations to the signal without affecting its characteristics nor making major changes in its shape. We include both types of augmentation to introduce variations of data, which increases model's generalization ability to perform well in unseen test set. We discuss more details about the augmentation selection in Section 5.6.

Formally, for each input sample x, we denote its strongly augmented view as x^s, and its weakly augmented view as x^w, where $x^s \sim \mathcal{T}_s$ and $x^w \sim \mathcal{T}_w$. These views are then passed to the encoder to extract their high dimensional latent representations. In particular, the encoder has a 3-block convolutional architecture as proposed in [30]. For an input $\mathbf{x}$, the encoder maps $\mathbf{x}$ into a high-dimensional latent representation $\mathbf{z} = f_{enc}(\mathbf{x})$. We define $\mathbf{z} = [z_1, z_2, \dots z_T]$, where T is the total timesteps, $z_i \in \mathbb{R}^d$, where d is the feature length. Thus, we get $\mathbf{z}^s$ for the strong augmented views, and $\mathbf{z}^w$ for the weak augmented views, which are then fed into the temporal contrasting module.

3.2 Temporal Contrasting

The Temporal Contrasting module deploys a contrastive loss to extract temporal features in the latent space with an autoregressive model. Given the latent representations $\mathbf{z}$, the autoregressive model f_{ar} summarizes all $\mathbf{z}_{\leq t}$ into a context vector $c_t = f_{ar}(\mathbf{z}_{\leq t})$, $c_t \in \mathbb{R}^h$, where h is the hidden dimension of f_{ar}. The context vector c_t is then used to predict the timesteps from z_{t+1} until z_{t+k} $(1 < k \leq K)$. To predict future timesteps, we use log-bilinear model that would preserve the mutual information between the input x_{t+k} and c_t, such that $f_k(x_{t+k}, c_t) = exp((\mathcal{W}_k(c_t))^T z_{t+k})$, where $\mathcal{W}_k$ is a linear function that maps c_t back into the same dimension as z, i.e., $\mathcal{W}_k : \mathbb{R}^{h \to d}$.

In our approach, the strong augmentation generates c_t^s and the weak augmentation generates c_t^w. We propose a tough cross-view prediction task by using the context of the strong augmentation c_t^s to predict the future timesteps of the weak augmentation z_{t+k}^w and vice versa. The contrastive loss tries to maximize the dot product between the predicted representation and the true one of the same sample, while minimizing the dot product with the other samples $\mathcal{N}_{t,k}$ within the mini-batch. Accordingly, we calculate the two losses $\mathcal{L}_{TC}^s$ and $\mathcal{L}_{TC}^w$ as follows:

$$\mathcal{L}_{TC}^s = -\frac{1}{K}\sum_{k=1}^{K} \log \frac{\exp((\mathcal{W}_k(c_t^s))^T z_{t+k}^w)}{\sum_{n \in \mathcal{N}_{t,k}} \exp((\mathcal{W}_k(c_t^s))^T z_n^w)} \quad (1)$$

$$\mathcal{L}_{TC}^w = -\frac{1}{K}\sum_{k=1}^{K} \log \frac{\exp((\mathcal{W}_k(c_t^w))^T z_{t+k}^s)}{\sum_{n \in \mathcal{N}_{t,k}} \exp((\mathcal{W}_k(c_t^w))^T z_n^s)} \quad (2)$$

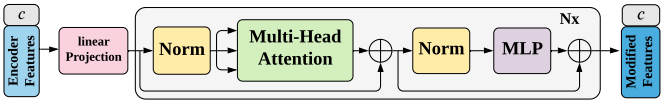


Fig. 3: Architecture of Transformer model used in Temporal Contrasting module. The token c in the output is sent next to the Contextual Contrasting module.

We use Transformer as the autoregressive model because of its efficiency and speed [31]. The architecture of the Transformer model is shown in Fig. 3. It mainly consists of multi-headed attention (MHA) followed by a Multilayer Perceptron (MLP) block. The MLP block is composed of two fully-connected layers with a non-linearity ReLU function and dropout in between. Pre-norm residual connections, which can produce more stable gradients [32], are adopted in our Transformer. We stack L identical layers to generate the final features. Inspired by BERT model [33], we add a token $c \in \mathbb{R}^h$ to the input, whose state acts as a representative context vector in the output. The operation of the Transformer starts by applying the features $\mathbf{z}_{\leq t}$ to a linear projection $\mathcal{W}_{Tran}$ layer that maps the features into the hidden dimension, i.e., $\mathcal{W}_{Tran} : \mathbb{R}^{d \to h}$. The output of this linear projection is then sent to the Transformer i.e., $\tilde{\mathbf{z}} = \mathcal{W}_{Tran}(\mathbf{z}_{\leq t})$, $\tilde{\mathbf{z}} \in \mathbb{R}^h$. Next, we attach the context vector into the feature vector $\tilde{\mathbf{z}}$ such that the input features become $\psi_0 = [c; \tilde{\mathbf{z}}]$, where the subscript 0 denotes being the input to the first layer. Next, we pass ψ_0 through Transformer layers as in the following equations:

$$\tilde{\psi_l} = \text{MHA}(\text{Norm}(\psi_{l-1})) + \psi_{l-1}, \qquad 1 \leq l \leq L; \quad (3)$$

$$\psi_l = \text{MLP}(\text{Norm}(\tilde{\psi_l})) + \tilde{\psi_l}, \qquad 1 \leq l \leq L. \quad (4)$$

Finally, we re-attach the context vector from the final output such that $c_t = \psi_L^0$. This context vector will be the input of the following contextual contrasting module.

3.3 Contextual Contrasting

We further propose a contextual contrasting module that aims to learn more discriminative representations. It starts with applying a non-linear transformation to the contexts using a non-linear projection head as in [4]. The projection head maps the contexts into the space where the contextual contrasting is applied.

Given a batch of N input samples, we will have two contexts for each sample from its two augmented views,

and thus have $2N$ contexts. For a context c_t^i, we denote $c_t^{i^+}$ as the positive sample of c_t^i that comes from the other augmented view of the same input, and hence, $(c_t^i, c_t^{i^+})$ is considered to be a positive pair. Meanwhile, the remaining $(2N-2)$ contexts from other inputs within the same batch are considered as the negative samples of c_t^i, i.e., c_t^i can form $(2N-2)$ negative pairs with its negative samples. Therefore, we can derive a contextual contrasting loss to maximize the similarity between the sample and its positive pair, while minimizing its similarity with the negative samples within the mini-batch. As such, the final representations can be discriminative.

Eq. 6 defines the contextual contrasting loss function $\mathcal{L}_{CC}$. Given a context c_t^i, we divide its similarity with its positive sample $c_t^{i^+}$ by its similarity with all the other $(2N-1)$ samples, including the positive pair and $(2N-2)$ negative pairs, to normalize the loss.

$$\ell(i,i^+) = -\log \frac{\exp\left(\text{sim}\left(\boldsymbol{c}_t^i, \boldsymbol{c}_t^{i^+}\right)/\tau\right)}{\sum_{m=1}^{2N} \mathbb{1}_{[m\neq i]} \exp\left(\text{sim}\left(\boldsymbol{c}_t^i, \boldsymbol{c}_t^m\right)/\tau\right)}, \quad (5)$$

$$\mathcal{L}_{CC} = \frac{1}{2N}\sum_{k=1}^{2N} \left[\ell(2k-1,2k) + \ell(2k,2k-1)\right], \quad (6)$$

where $sim(\boldsymbol{u},\boldsymbol{v}) = \boldsymbol{u}^T\boldsymbol{v}/\|\boldsymbol{u}\|\|\boldsymbol{v}\|$ denotes the dot product between ℓ_2 normalized $\boldsymbol{u}$ and $\boldsymbol{v}$ (i.e., cosine similarity), $\mathbb{1}_{[m\neq i]} \in \{0,1\}$ is an indicator function, evaluating to 1 iff $m \neq i$, and τ is a temperature parameter.

The overall self-supervised loss is the combination of the two temporal contrasting losses and the contextual contrasting loss as follows.

$$\mathcal{L}_{unsup} = \lambda_1 \cdot (\mathcal{L}_{TC}^s + \mathcal{L}_{TC}^w) + \lambda_2 \cdot \mathcal{L}_{CC}, \quad (7)$$

where λ_1 and λ_2 are fixed scalar hyperparameters denoting the relative weight of each loss.

3.4 Class-Aware TS-TCC

We propose a second variant of our framework provided the existence of few labeled data to further improve the representation learned by TS-TCC. In this variant, we aim to overcome one drawback in the contextual contrasting module i.e., considering all the samples in mini-batch as negative pairs, even if they belong to the same class of the anchor sample. Therefore, we replace the contextual contrasting with a supervised contextual contrasting to consider the class information while selecting the positive and negative pairs. We call this variant as Class-Aware TS-TCC (CA-TCC).

Supervised contrastive learning was first proposed to improve the supervised cross-entropy loss [34]. However, we reuse it in our framework to improve the contextual contrasting discussed in Sec. 3.3, by leveraging the label information throughout the training process. Specifically, supervised contrastive pulls the embeddings from the similar classes together, while pushing them from the dissimilar classes away within the mini-batch. Therefore, instead of having a single positive pair (from augmented views), we use multiple instances from the same class as positive pairs. In the same way, all the other samples from different classes are considered as negative samples. Having many instances of positive pairs was found to improve the learned representations [34].

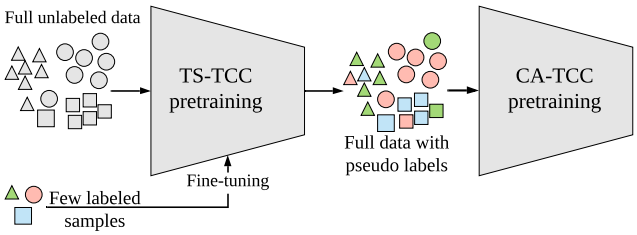


Fig. 4: CA-TCC training process. First, TS-TCC is trained with fully unlabeled data, then we use the available few labeled samples to fine-tune the pretrained model. The fine-tuned TS-TCC model is then used to generate pseudo labels for the unlabeled data. Finally, the full data with pseudo labels are used to train CA-TCC.

However, applying supervised contextual contrasting requires the availability of the full labeled data, which is impractical in many real-world scenarios, where only few labeled samples are usually available. Therefore, in CA-TCC, we make use of the available few labeled samples to fine-tune the pretrained TS-TCC. Next, the fine-tuned model is used to generate pseudo labels for the unlabeled data. Finally, we use these pseudo labels to train our CA-TCC. This process is further explained in Fig. 4.

Formally, assuming that the dataset consists of N labeled samples $\{\mathbf{x}_k, y_k\}_{k=1\ldots N}$, then after applying augmentations, the dataset becomes of $2N$ samples, $\{\hat{\mathbf{x}}_l, \hat{y}_l\}_{l=1\ldots 2N}$ such that $\hat{\mathbf{x}}_{2k}$ and $\hat{\mathbf{x}}_{2k-1}$ are the two views of $\mathbf{x}_k$, and similarly $y_k = \hat{y}_{2k} = \hat{y}_{2k-1}$. Also assuming that $i \in I \equiv \{1\ldots 2N\}$ represents the index of an arbitrary augmented sample, and $A(i) \equiv I \setminus \{i\}$, the supervised contextual contrasting loss can be expressed as:

$$\mathcal{L}_{SCC} = \sum_{i\in I} \frac{1}{|P(i)|} \sum_{p\in P(i)} \ell(i,p), \quad (8)$$

$$P(i) = \{p \in A(i) : \hat{y}_p = \hat{y}_i\}, \quad (9)$$

where $P(i)$ is the set of indices of all samples with the same class as the sample $\hat{\mathbf{x}}_i$ in the batch, and (i,p) for any $p \in P(i)$ is thus a positive pair. $|P(i)|$ is the cardinality of $P(i)$. Thus, the overall loss can be expressed as:

$$\mathcal{L}_{semi} = \lambda_3 \cdot (\mathcal{L}_{TC}^s + \mathcal{L}_{TC}^w) + \lambda_4 \cdot \mathcal{L}_{SCC}, \quad (10)$$

where λ_3 and λ_4 are fixed scalar hyperparameters denoting the relative weight of each loss.

4 Experimental Setup

4.1 Datasets

To comprehensively evaluate our proposed models, we adopted three publicly available real-world datasets for human activity recognition, sleep stage classification and epileptic seizure prediction tasks. Additionally, we investigated the transferability of our learned features on a fault diagnosis dataset.

4.1.1 Human Activity Recognition

We employed UCI HAR dataset[1] [35], which contains sensor readings for 30 subjects performing 6 activities (i.e., walking, walking upstairs, downstairs, standing, sitting, and lying down). The data were collected by a mounted Samsung Galaxy S2 device on users' waist, with a sampling rate of 50 Hz.

4.1.2 Sleep Stage Classification

We adopted Sleep-EDF dataset[2] [36], which includes whole-night polysomnography (PSG) sleep recordings for 20 subjects. In particular, each recording contains two electroencephalogram (EEG) channels namely Fpz-Cz and Pz-Oz, with a sampling rate of 100 Hz. In this work, we used a single EEG channel (i.e., Fpz-Cz), following previous studies [37], [38]. Sleep stage classification refers to classifying the input EEG signal into one of five classes: Wake (W), Non-rapid eye movement (N1, N2, N3) and Rapid Eye Movement (REM).

4.1.3 Epilepsy Seizure Prediction

The Epileptic Seizure Recognition dataset[3] [39] consists of EEG recordings from 500 subjects, where the brain activity was recorded for each subject for 23.6 seconds. The original dataset is labeled with five classes, but as four of them do not include epileptic seizures, we merged them into one class and treat it as a binary classification problem.

4.1.4 Fault Diagnosis (FD)

Fault Diagnosis dataset[4] [40] was collected from sensor readings of bearing machine under four different working conditions. Each working condition can be considered as a separate domain as it has different characteristics from the other working conditions (e.g., rotational speed and load torque) [41]. Each domain has three classes; two fault classes (i.e., inner fault and outer fault) and one healthy class. We use this dataset to conduct the transferability experiment and show the effectiveness of our method in transfer learning scenarios.

TABLE 1: Description of datasets used in our experiments. The details of FD is the same for all the 4 working conditions.

Dataset	# Train	# Test	Length	# Channel	# Class
HAR	7,352	2,947	128	9	6
Sleep-EDF	25,612	8,910	3,000	1	5
Epilepsy	9,200	2,300	178	1	2
FD	8,184	2,728	5,120	1	3

Table 1 summarizes the details of each dataset, i.e., the number of training samples (# Train) and testing samples (# Test), the length of the sample, the number of sensor channels (# Channel) and the number of classes (# Class).

1. https://archive.ics.uci.edu/ml/datasets/human+activity+recognition+using+smartphones
2. https://physionet.org/physiobank/database/sleep-edfx/
3. https://archive.ics.uci.edu/ml/datasets/Epileptic+Seizure+Recognition
4. https://mb.uni-paderborn.de/en/kat/main-research/datacenter/bearing-datacenter/data-sets-and-download

4.2 Implementation Details

We split the data into 60%, 20%, 20% for training, validation and testing, with considering subject-wise split for Sleep-EDF dataset to avoid overfitting. Experiments were repeated for 5 times with 5 different seeds, and we reported the performance with mean and standard deviation. The pretraining and downstream tasks were done for 40 epochs, as we noticed that the performance does not improve with further training. We applied a batch size of 128 (which was reduced to 32 in *few-labeled data* experiments as data size may be less than 128). We used Adam optimizer with a learning rate of 3e-4, weight decay of 3e-4, $\beta_1 = 0.9$, and $\beta_2 = 0.99$. For the strong augmentation, we set the number of segments M as: $M_{HAR} = 10,\ M_{Ep} = 12$ and $M_{EDF} = 20$ (see Section 5.6), while for the weak augmentation, we set the scaling ratio to 2. In TS-TCC, we set $\lambda_1 = 1$, while we achieved good performance when λ_2 is around 1. Particularly, we set λ_2 as 0.7 in our experiments. For CA-TCC, was set $\lambda_3 = 0.01$ and $\lambda_4 = 0.7$ (see Section 5.5). In the Transformer, we set the $L = 4$, and the number of heads as 4. We tuned $h \in \{32, 50, 64, 100, 128, 200, 256\}$ and set $h_{HAR,Ep} = 100$, $h_{EDF} = 64$. We also set its dropout to 0.1. In contextual contrasting, we set $\tau = 0.2$. Lastly, we built our model using PyTorch 1.7 and trained it on a NVIDIA GeForce RTX 2080 Ti GPU.

5 Experimental Results

To show the efficacy of our proposed TS-TCC and CA-TCC, we evaluate them on three different training settings, including linear evaluation, semi-supervised training and transfer learning. We measure the performance using two metrics namely the accuracy and the macro-averaged F1-score (MF1) to better verify on the imbalanced datasets.

5.1 Comparison with Baseline Approaches

We compare our proposed TS-TCC and CA-TCC against the following baselines. (1) **Random Initialization**: training a linear classifier on top of frozen and randomly initialized encoder, and this represents the lower bound; (2) **Supervised**: supervised training of both encoder and classifier; (3) **SSL-ECG** [26]; (4) **CPC** [21]; (5) **SimCLR** [4]; and (6) **FixMatch** [24]. It is worth noting that, we use our time-series specific augmentations to pretrain SimCLR, as it was originally designed for images. Similarly, as FixMatch relies on weak and strong augmentations in training, we trained it using our proposed augmentations.

To evaluate the performance of SSL-ECG, CPC, SimCLR and TS-TCC, we first pretrain them without any labeled data. Next, we use a portion of the labeled data to evaluate their performance. We follow the standard *linear evaluation* scheme [4], [21], in which we train a linear classifier (single fully connected layer) on top of a *frozen* self-supervised pretrained encoder model. Notably, only FixMatch and CA-TCC use some few labeled data during pretraining, as they are semi-supervised approaches. Table 2 shows the linear evaluation results of our approaches against the baseline methods using 1% and 5% of the labeled data.

In overall, our proposed TS-TCC, performs best on the three datasets compared to the supervised training and the

TABLE 2: Comparison between our proposed TS-TCC and CA-TCC against baselines using *linear evaluation* experiment using 1% and 5% of labeled data. Methods indicated by * included the labeled data in pretraining. Best results are in bold, and the second bests are underlined.

1% of labeled data						
	HAR		Sleep-EDF		Epilepsy	
Baseline	ACC	MF1	ACC	MF1	ACC	MF1
Random Initialization	39.81±3.77	34.67±5.04	20.54±0.96	20.73±1.39	70.31±2.11	66.15±2.55
Supervised	44.93±6.66	41.02±6.75	34.10±0.28	30.82±0.77	76.09±0.68	74.79±0.42
SSL-ECG [26]	60.02±3.96	54.04±6.00	70.94±0.83	61.59±0.66	89.33±0.43	85.99±0.31
CPC [21]	65.41±1.42	63.77±1.70	74.71±0.19	68.74±0.07	88.95±1.14	85.80±0.30
SimCLR [4]	65.81±0.69	64.25±0.92	57.99±1.02	56.91±0.61	88.27±1.45	84.01±0.97
TS-TCC *(Ours)*	70.48±0.33	69.46±0.48	75.81±0.39	69.98±0.24	91.17±0.53	89.17±0.15
FixMatch* [24]	70.07±0.87	55.35±1.96	42.83±3.88	33.23±1.96	89.03±3.84	81.17±1.52
CA-TCC* *(Ours)*	**77.29±0.57**	**76.23±0.12**	**79.39±0.09**	**70.76±0.52**	**92.03±0.09**	**91.88±0.14**
5% of labeled data						
	HAR		Sleep-EDF		Epilepsy	
Baseline	ACC	MF1	ACC	MF1	ACC	MF1
Random Initialization	49.64±2.46	45.76±1.99	22.82±2.80	22.75±2.18	75.52±3.56	70.51±3.34
Supervised	52.80±1.53	50.93±0.24	60.51±3.91	54.76±5.48	83.42±0.67	80.43±0.75
SSL-ECG [26]	63.70±5.34	58.62±7.38	73.36±0.46	63.68±0.12	92.77±0.22	89.02±0.34
CPC [21]	75.39±2.14	74.67±2.45	76.30±0.42	70.49±0.26	92.83±0.31	90.20±0.53
SimCLR [4]	75.82±1.36	74.91±1.51	64.15±1.01	61.85±0.78	91.25±0.51	89.24±0.96
TS-TCC *(Ours)*	77.58±1.78	76.66±1.96	76.98±0.56	70.94±0.46	93.12±0.31	93.67±0.56
FixMatch* [24]	**90.71±0.55**	80.04±0.93	65.46±3.95	39.22±1.93	94.11±1.44	91.17±0.32
CA-TCC* *(Ours)*	88.27±0.38	**88.29±0.34**	**82.14±0.19**	**74.75±0.06**	**94.52±0.12**	**94.00±0.09**

self-supervised methods. This demonstrates the powerful representation learning capability of our TS-TCC model. It is worth noting that contrastive methods (e.g., CPC, SimCLR and our TS-TCC) generally achieve better results than the pretext-based method (i.e., SSL-ECG), which reflects the power of invariant features learned by contrastive methods. Additionally, CPC method shows better results than SimCLR, indicating that temporal features are more important than general features in time-series data. We notice that our CA-TCC achieves the best performance in all the datasets with a wide margin, especially in the scenario with 1% of labeled data. This shows its effectiveness in taking more advantage of the available few labels over the other semi-supervised method, FixMatch, to learn useful representations throughout the different datasets.

5.2 TS-TCC Semi-Supervised Experiments

Here, we investigate our TS-TCC under different semi-supervised settings, by fine-tuning the pretrained model using 1%, 5%, 10%, 50%, and 75% of randomly selected instances of the training data. Fig. 5 shows the results of our fine-tuned TS-TCC along with the supervised training under the aforementioned settings. We evaluate the performance with MF1 because of the imbalance in the Sleep-EDF dataset. In particular, TS-TCC fine-tuning (i.e., red curves in Fig. 5) means that we fine-tune the pretrained encoder with the few labeled samples.

We observe that supervised training performs poorly with limited labeled data, while our TS-TCC fine-tuning achieves significantly better performance than supervised training. For example, with only 1% of labeled data, TS-TCC fine-tuning can still achieve 69.5% and 89.2% for HAR and Epilepsy datasets respectively, compared with 47.6% and 74.8% for the supervised training on the same datasets. Furthermore, with only 10% of labeled data, our fine-tuned TS-TCC can achieve comparable performance with the supervised training with 100% of labeled data in the three datasets, demonstrating its effectiveness under

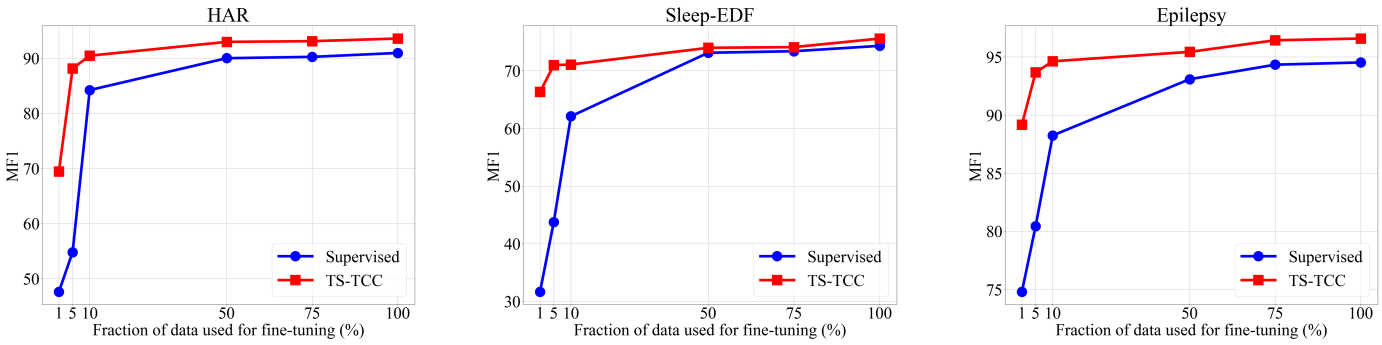


Fig. 5: Comparison between supervised training vs. TS-TCC *fine-tuning* with different few labeled data scenarios in terms of MF1.

TABLE 3: Cross-domain transfer learning experiments performed on Fault Diagnosis dataset in terms of accuracy. For CA-TCC, we used 1% of labeled data for pretraining. Best results are in bold and second bests are underlined.

Method	A→B	A→C	A→D	B→A	B→C	B→D	C→A	C→B	C→D	D→A	D→B	D→C	AVG
Supervised	34.38	44.94	34.57	**52.93**	63.67	<u>99.82</u>	**52.93**	84.02	83.54	**53.15**	99.56	62.43	63.83
TS-TCC	<u>43.15</u>	<u>51.50</u>	<u>42.74</u>	47.98	<u>70.38</u>	99.30	38.89	<u>98.31</u>	<u>99.38</u>	<u>51.91</u>	<u>99.96</u>	<u>70.31</u>	<u>67.82</u>
CA-TCC	**44.75**	**52.09**	**45.63**	<u>46.26</u>	**71.33**	**100.0**	<u>52.71</u>	**99.85**	**99.84**	46.48	**100.0**	**77.01**	**69.66**

TABLE 4: Ablation study of each component in CA-TCC model performed with *linear evaluation* experiment with 5% of labeled data.

	HAR		Sleep-EDF		Epilepsy	
Component	ACC	MF1	ACC	MF1	ACC	MF1
TC only	68.16±1.15	66.89±1.11	75.55±0.93	60.19±0.81	88.29±1.29	88.00±1.91
TC + X-Aug	74.22±1.03	72.18±0.99	77.80±0.29	61.28±1.22	90.51±0.43	89.27±0.22
TS-TCC (TC + X-Aug + CC)	77.58±1.78	76.66±1.96	76.98±0.56	70.94±0.46	93.12±0.31	93.67±0.56
CA-TCC (TC + X-Aug + SCC)	**88.27±0.38**	**88.29±0.34**	**82.14±0.19**	**74.75±0.06**	**94.52±0.12**	**94.00±0.09**
TS-TCC (Weak only)	67.39±1.73	65.54±2.42	79.63±0.16	68.15±0.23	93.22±0.11	91.97±0.19
CA-TCC (Weak only)	85.68±0.26	84.77±0.25	81.62±0.89	70.10±1.28	93.84±0.05	92.19±0.10
TS-TCC (Strong only)	50.37±1.18	43.05±1.42	74.84±0.50	64.53±0.58	92.49±0.62	90.60±0.20
CA-TCC (Strong only)	59.59±0.06	53.34±0.49	79.24±0.74	69.39±0.89	93.74±0.04	92.00±0.05

different semi-supervised settings.

As shown in Fig. 5, TS-TCC fine-tuning is able to achieve impressive performance with few labeled data. Hence, TS-TCC is expected to generate high quality pseudo labels in Fig. 4, which motivates us to propose the variant CA-TCC.

5.3 Transfer Learning Experiment

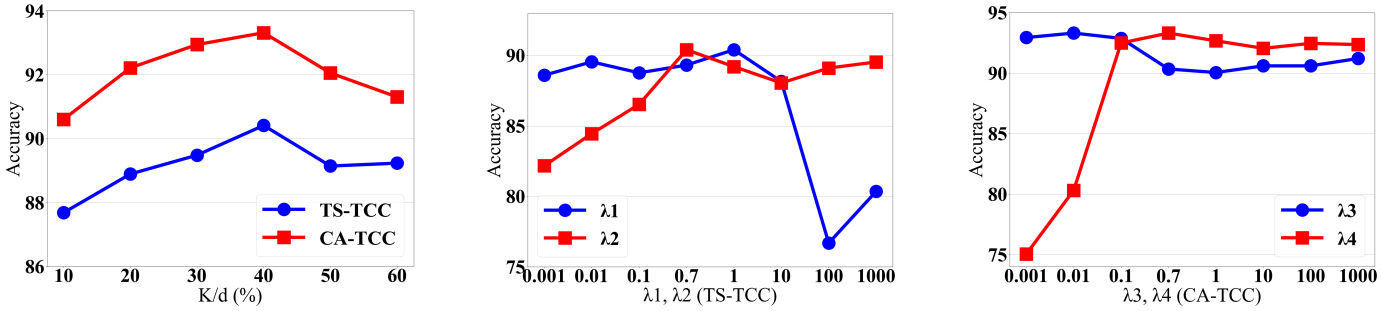
We further examine the transferability of the learned features by designing a transfer learning experiment. We use Fault Diagnosis (FD) dataset for the evaluation under the transfer learning setting. Recall that FD dataset has four working conditions, which are considered as four domains (denoted as domains A, B, C and D). Here, we train the model on the data from one condition (i.e., source domain) and test it on another condition (i.e., target domain). We adopt three training schemes on the source domain, namely, (1) Supervised training, (2) TS-TCC fine-tuning and (3) CA-TCC fine-tuning. In TS-TCC and CA-TCC fine-tuning, we fine-tune our pretrained encoder using the labeled data in the source domain.

Table 3 shows the performance of the three training schemes under 12 cross-domain scenarios. Clearly, our pretrained TS-TCC model consistently outperforms the supervised pretraining in 8 out of 12 cross-domain scenarios. Similarly, with only 1% of labels in each source domain for training, we find that CA-TCC model outperforms the supervised pretraining in 9 out of 12 cross-domain scenarios. We find that TS-TCC model can achieve at least ∼7% improvement in 7 out of 8 winning scenarios (except for D→B scenario). Similarly, CA-TCC model can achieve at least ∼8% improvement in 7 out of 9 winning scenarios. Overall, our two proposed approaches can improve the transferability of learned representations over the supervised training by ∼ 4% and 6% in terms of accuracy.

5.4 Ablation Study

We study the effectiveness of each component in our proposed CA-TCC model. Specifically, we derive different model variants for comparison as follows. First, we train the Temporal Contrasting (TC) module without the cross-view prediction task, where each branch predicts the future time-steps of the same augmented view. This variant is denoted as 'TC only'. Second, we train the TC module with adding the cross-view prediction task, which is denoted as 'TC + X-Aug'. Third, we train the proposed TS-TCC model, which is denoted as 'TC + X-Aug + CC'. Finally, we train the proposed CA-TCC model, which is denoted as 'TC + X-Aug + SCC'. We also study the effect of using a single family of augmentations on the performance of TS-TCC and CA-TCC. In particular, for an input x, we generate two different views x_1 and x_2 from the same augmentation type, i.e., $x_1 \sim \mathcal{T}_w$ and $x_2 \sim \mathcal{T}_w$ when using either the weak augmentation or the strong augmentation. We show the linear evaluation results in terms of accuracy and macro F1-score with only 5% throughout these experiments.

Table 4 shows this ablation study on the three datasets. Clearly, the proposed cross-view prediction task generates robust features and thus improves the performance by more than 6% accuracy on HAR datasets, and ∼2% on Sleep-EDF and Epilepsy datasets. Additionally, the contextual contrasting module further improves the performance, as it helps the features to be more discriminative. More improvement was achieved by using supervised contextual contrasting in CA-TCC, which supports the importance of considering more positive samples from the same class to generate more discriminative features. Studying the augmentations effect on TS-TCC, we find that generating different views from the same augmentation type is not helpful with HAR and Sleep-EDF datasets. For these complex datasets, using only weak augmentations may not make a *tough* cross-view prediction task, leading to close results to the variant 'TC only'. Counterpart, using only strong augmentations deviates the model from recognizing the original data while testing. However, the less complex Epilepsy dataset can still achieve comparable performance with only one augmentation. For CA-TCC, we find that it consistently outperforms the results of TS-TCC, showing its effectiveness to improve the



(a) (b) (c)

Fig. 6: Sensitivity analysis experiments on HAR dataset. Figure (a) shows the selection of percentage of future timesteps. Figure (b) shows the impacts of λ_1, λ_2 in TS-TCC, while figure (c) shows the effects of varying λ_3 and λ_4 in CA-TCC.

representations with the available few labeled samples. For example, we find that it highly improves the performance of using only weak or strong augmentations in both HAR and Sleep-EDF datasets.

5.5 Sensitivity Analysis

We perform sensitivity analysis on HAR dataset to study five parameters namely, the number of predicted future timesteps K in the temporal contrasting module, λ_1 and λ_2 in Eq. 7, and λ_3 and λ_4 in Eq. 10. In specific, we used linear evaluation experiment with full labels to assess the performance.

Fig. 6a shows the effect of K on the overall performance of TS-TCC and CA-TCC, where x-axis is the percentage K/d, and d is the length of the features. Clearly, increasing the percentage of the predicted future timesteps improves the performance. However, larger percentages can harm the performance as it reduces the amount of past data used for training the autoregressive model. We observe that predicting 40% of the total feature length performs the best, and thus we set K as $d\times40\%$ in our experiments. The same conclusion can be drawn for both variants of our framework. Fig. 6b shows the results of varying λ_1 and λ_2 in TS-TCC (Eq. 7) in a range between 0.001 and 1000 respectively. We first fix $\lambda_1 = 1$ and change the values of λ_2. We observe that our model achieves good performance when $\lambda_2 \approx 1$, where the model performs best with $\lambda_2 = 0.7$. Consequently, we fix $\lambda_2 = 0.7$ and tune the value of λ_1 as in Fig. 6b, where we find that our model achieves the best performance when $\lambda_1 = 1$. We also find that as $\lambda_1 < 10$, our model is less sensitive to its value, while it is more sensitive to different values of λ_2.

We also perform sensitivity analysis on the values of λ_3 and λ_4 in CA-TCC (Eq. 10) in a similar manner and within the same ranges, as shown in Fig. 6c. We also used 1% of labels to generate pseudo labels. Consequently, we chose the values of $\lambda_3 = 0.01$ and $\lambda_4 = 0.7$.

5.6 Augmentation Selection

The proper selection of augmentations is crucial for contrastive learning techniques, as contrastive methods are sensitive to the choice of augmentations [4]. Many studies showed that composing multiple data augmentation operations achieves better performance for image data [4], [34].

TABLE 5: A study of TS-TCC *linear evaluation* performance with 5% of labeled HAR data when using different variations of weak and strong augmentations.

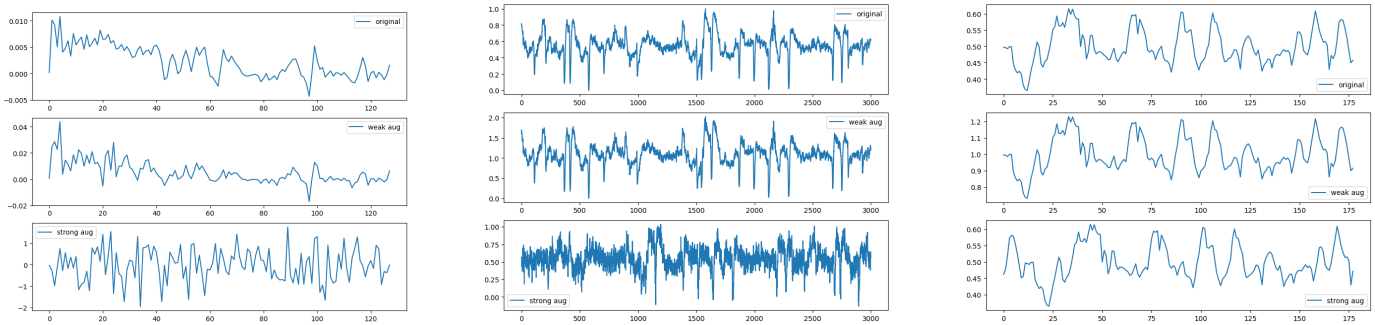
Weak Augmentation	Strong Augmentation	ACC	MF1
scale	no Aug	46.12	36.67
scale + jitter	no Aug	56.98	50.88
no Aug	permutation	62.07	52.64
no Aug	jitter + permutation	72.35	68.03
time Shift	jitter + permutation	71.57	66.97
time Shift + jitter	jitter + permutation	74.69	69.33
scale	jitter + permutation	72.59	68.90
scale + jitter	jitter + permutation	**77.58**	**76.66**

However, the selection of the proper augmentations that well fits time-series data is less explored and still an open problem [42]. Here, we aim to study the choice of the suitable augmentations for our contrastive learning problem.

We define the weak augmentation as the one that applies limited change on the shape of the original signal as shown in the second row of Fig. 7. Examples for weak augmentations include scaling and time shifting. On the other hand, strong augmentation makes strong perturbations on the signal shape with keeping some of its temporal information, such as permutation. This is shown in the third row of Fig. 7. Permutation includes splitting the signal into M chunks and shuffling their order. Notably, some augmentations such as jittering (i.e., adding random noise) can be considered as both strong and weak augmentations depending on the added noise level. To justify the selection of our proposed augmentations, we provide a systematic study on the impact of applying several different data augmentations for each view in TS-TCC (Fig. 2). The results are provided in Table 5.

We first apply weak augmentation only, i.e., scaling, to one view without applying any augmentation to the other view, and the accuracy is only 46.12% as shown in Table 5. However, by composing jitter to scaling, we can observe a large performance improvement. Meanwhile, we also apply strong augmentation only and keep the other view without augmentations. The accuracy in this case is 62.07%, which is much higher than applying only weak augmentation. Besides, we notice that adding jitter to the permutation can further improve the accuracy to 72.35%.

We also test to apply another weak augmentation that does not highly affect the signal characteristics along with



(a) HAR (b) Sleep-EDF (c) Epilepsy

Fig. 7: Sample from each adopted dataset after normalization along with its weak and strong augmented views. The first row is the original samples, the second row shows the weak augmented views, while the third row shows the strong augmented views.

the strong augmentation. We can find that using time shift as a weak augmentation achieves relatively good performance, i.e., an accuracy of 71.57%. Again, it can be found that adding jitter to time shift further improves the accuracy to 74.69%. Similar results are achieved when replacing time shift with scaling. In addition, applying only weak or strong augmentations to both views results in poorer performance as shown in Table 4. At this point, we find that applying weak augmentation for one view and strong augmentations for another views achieves the best performance, i.e., an accuracy of 77.58%. As an explanation, weak augmentations apply limited changes on the shape of the original signal as shown in Fig. 7, which helps the model to perform well on the test data.

Via observing the different characteristics of three different samples in Fig. 7 in terms of signal magnitude and sampling rates, we conclude that the choice of proper parameters for augmentations (e.g., jitter and scaling ratio) will vary from one dataset to another. Consequently, the range of parameter choices will be highly depending on the characteristics of each time-series data. Therefore, we propose to normalize the signals as a preprocessing step in our framework to improve parameter selection. For example, the value of added jitter after normalizing the signals in weak augmentation should be less than the ones added in the strong augmentation. In the experiments, we observe that the best practice is to normalize the data between 0 and 1 and set the weak-jittering to be in the range [0, 0.1] and the strong jitter in the range [0.1, 1]. Similarly, a scaling ratio of 2 would be sufficient for the weak augmentation in any time-series signals.

Similarly, for the strong augmentation, it is important to properly select the number of chunks M, where the value of M in time-series data with longer sequences should be greater than its value in those with shorter sequences. We find that selecting 40% of the total feature size achieves the best performance for the three time-series datasets (see Section 5.5).

6 Conclusion

In this paper, we proposed a novel semi-supervised framework called CA-TCC for time-series data classification. CA-TCC relies on our self-supervised framework TS-TCC that learns powerful representations from unlabeled data. Specifically, we propose time-series specific weak and strong augmentations and provide a systematic study to the choice of these augmentations. We use these augmentations to learn transformation-invariant representations in our proposed temporal and contextual contrasting modules. By training a linear classifier with few labels on top of the learned representation by TS-TCC, it achieved a comparable performance to the fully-supervised training. However, one drawback of the contextual contrasting is considering samples from same class as negative samples, which might limit the performance. Therefore, we propose CA-TCC in the semi-supervised settings to consider the labels information while choosing the positive and negative pairs. We find that the linear evaluation of our TS-TCC by using only 10% of the labeled data can achieve close performance to the supervised training with full labeled data. In addition, CA-TCC was able to further improve this performance with only 1% of labeled data when testing on three different datasets. In addition, both variants were able to improve the transferability of the representations in real-world transfer learning scenarios.

References

[1] Arash Gharehbaghi and Maria Lindén. A deep machine learning method for classifying cyclic time series of biological signals using time-growing neural network. *IEEE transactions on neural networks and learning systems*, 29(9):4102–4115, 2017.

[2] Travers Ching, Daniel S Himmelstein, Brett K Beaulieu-Jones, Alexandr A Kalinin, Brian T Do, Gregory P Way, Enrico Ferrero, Paul-Michael Agapow, Michael Zietz, Michael M Hoffman, et al. Opportunities and obstacles for deep learning in biology and medicine. *Journal of The Royal Society Interface*, 2018.

[3] Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *European Conference on Computer Vision*, pages 69–84, 2016.

[4] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *ICML 2020: 37th International Conference on Machine Learning*, volume 1, pages 1597–1607, 2020.

[5] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. In *International Conference on Learning Representations*, 2018.

[6] R. Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Philip Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. In *International Conference on Learning Representations*, 2018.

[7] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9729–9738, 2020.

[8] Jean-Yves Franceschi, Aymeric Dieuleveut, and Martin Jaggi. Unsupervised scalable representation learning for multivariate time series. In *Conference on Neural Information Processing Systems (NeurIPS)*, volume 32, pages 4650–4661, 2019.

[9] Mostafa Neo Mohsenvand, Mohammad Rasool Izadi, and Pattie Maes. Contrastive representation learning for electroencephalogram classification. In *Machine Learning for Health NeurIPS Workshop*, 2020.

[10] Joseph Y Cheng, Hanlin Goh, Kaan Dogrusoz, Oncel Tuzel, and Erdrin Azemi. Subject-aware contrastive learning for biosignals. *arXiv preprint arXiv:2007.04871*, 2020.

[11] Emadeldeen Eldele, Mohamed Ragab, Zhenghua Chen, Min Wu, Chee Keong Kwoh, Xiaoli Li, and Cuntai Guan. Time-series representation learning via temporal and contextual contrasting. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 2352–2359, 2021.

[12] Guo-Jun Qi and Jiebo Luo. Small data challenges in big data era: A survey of recent progress on unsupervised and semi-supervised methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1, 2020.

[13] Longlong Jing and Yingli Tian. Self-supervised visual feature learning with deep neural networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1, 2020.

[14] Ashish Jaiswal, Ashwin Ramesh Babu, Mohammad Zaki Zadeh, Debapriya Banerjee, and Fillia Makedon. A survey on contrastive self-supervised learning. *Technologies*, 9(1):2, 2020.

[15] Richard Zhang, Phillip Isola, and Alexei A. Efros. Colorful image colorization. In *European Conference on Computer Vision*, pages 649–666, 2016.

[16] Ishan Misra and Laurens van der Maaten. Self-supervised learning of pretext-invariant representations. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6707–6717, 2020.

[17] Xiaohua Zhai, Avital Oliver, Alexander Kolesnikov, and Lucas Beyer. S4l: Self-supervised semi-supervised learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[18] Nitish Srivastava, Elman Mansimov, and Ruslan Salakhudinov. Unsupervised learning of video representations using lstms. In *Proceedings of The 32nd International Conference on Machine Learning*, pages 843–852, 2015.

[19] Donglai Wei, Joseph Lim, Andrew Zisserman, and William T Freeman. Learning and using the arrow of time. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8052–8060, 2018.

[20] Xialei Liu, Joost van de Weijer, and Andrew D. Bagdanov. Exploiting unlabeled data in cnns by self-supervised learning to rank. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(8):1862–1878, 2019.

[21] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv: Learning*, 2018.

[22] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Koray Kavukcuoglu, Rémi Munos, and Michal Valko. Bootstrap your own latent: A new approach to self-supervised learning. In *Advances in Neural Information Processing Systems*, volume 33, pages 21271–21284, 2020.

[23] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15750–15758, 2020.

[24] Kihyuk Sohn, David Berthelot, Nicholas Carlini, Zizhao Zhang, Han Zhang, Colin A. Raffel, Ekin Dogus Cubuk, Alexey Kurakin, and Chun-Liang Li. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. In *Advances in Neural Information Processing Systems*, volume 33, pages 596–608, 2020.

[25] Aaqib Saeed, Tanir Ozcelebi, and Johan Lukkien. Multi-task self-supervised learning for human activity detection. *ACM Interact. Mob. Wearable Ubiquitous Technol.*, 2019.

[26] P. Sarkar and A. Etemad. Self-supervised ecg representation learning for emotion recognition. *IEEE Transactions on Affective Computing*, 2020.

[27] Aaqib Saeed, Victor Ungureanu, and Beat Gfeller. Sense and learn: Self-supervision for omnipresent sensors. *arXiv preprint arXiv:2009.13233*, 2020.

[28] Karan Aggarwal, Shafiq R. Joty, Luis Fernández-Luque, and Jaideep Srivastava. Adversarial unsupervised representation learning for activity time-series. In *33rd AAAI Conference on Artificial Intelligence, AAAI 2019, 31st Annual Conference on Innovative Applications of Artificial Intelligence, IAAI 2019 and the 9th AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019*, pages 834–841, 2019.

[29] Hubert J. Banville, Omar Chehab, Aapo Hyvärinen, Denis-Alexander Engemann, and Alexandre Gramfort. Uncovering the structure of clinical eeg signals with self-supervised learning. *Journal of Neural Engineering*, 18(4):46020, 2021.

[30] Zhiguang Wang, Weizhong Yan, and Tim Oates. Time series classification from scratch with deep neural networks: A strong baseline. In *International Joint Conference on Neural Networks (IJCNN)*, pages 1578–1585, 2017.

[31] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, volume 30, pages 5998–6008, 2017.

[32] Qiang Wang, Bei Li, Tong Xiao, Jingbo Zhu, Changliang Li, Derek F. Wong, and Lidia S. Chao. Learning deep transformer models for machine translation. In *ACL*, 2019.

[33] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina N. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Association for Computational Linguistics*, pages 4171–4186, 2018.

[34] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. In *Advances in Neural Information Processing Systems*, volume 33, pages 18661–18673, 2020.

[35] Davide Anguita, Alessandro Ghio, Luca Oneto, Xavier Parra, and Jorge Luis Reyes-Ortiz. A public domain dataset for human activity recognition using smartphones. In *European Symposium on Artificial Neural Networks*, pages 437–442, 2013.

[36] Ary L. Goldberger, Luis A. N. Amaral, Leon Glass, Jeffrey M. Hausdorff, Plamen Ch. Ivanov, Roger G. Mark, Joseph E. Mietus, George B. Moody, Chung-Kang Peng, and H. Eugene Stanley. Physiobank, physiotoolkit, and physionet components of a new research resource for complex physiologic signals. *Circulation*, 101(23):215–220, 2000.

[37] A. Supratak, H. Dong, C. Wu, and Y. Guo. Deepsleepnet: a model for automatic sleep stage scoring based on raw single-channel eeg. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 2017.

[38] Emadeldeen Eldele, Zhenghua Chen, Chengyu Liu, Min Wu, Chee-Keong Kwoh, Xiaoli Li, and Cuntai Guan. An attention-based deep learning approach for sleep stage classification with single-channel eeg. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 29:809–818, 2021.

[39] Ralph G. Andrzejak, Klaus Lehnertz, Florian Mormann, and et al. Rieke. Indications of nonlinear deterministic and finite-dimensional structures in time series of brain electrical activity: Dependence on recording region and brain state. *Phys. Rev. E*, 2001.

[40] Christian Lessmeier, James Kuria Kimotho, Detmar Zimmer, and Walter Sextro. Condition monitoring of bearing damage in electromechanical drive systems by using motor current signals of electric motors: A benchmark data set for data-driven classification. In *European conference of the prognostics and health management society*, 2016.

[41] Mohamed Ragab, Zhenghua Chen, Min Wu, Haoliang Li, Chee-Keong Kwoh, Ruqiang Yan, and Xiaoli Li. Adversarial multiple-target domain adaptation for fault classification. *IEEE Transactions on Instrumentation and Measurement*, 2020.

[42] Qingsong Wen, Liang Sun, Fan Yang, Xiaomin Song, Jingkun Gao, Xue Wang, and Huan Xu. Time series data augmentation for deep learning: A survey. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, 2021.