Contrast Everything: A Hierarchical Contrastive Framework for Medical Time-Series

Yihe Wang *
University of North Carolina - Charlotte
ywang145@uncc.edu

Yu Han*
University of Chinese Academy of Sciences
hanyu21@mails.ucas.ac.cn

Haishuai Wang
Zhejiang University
haishuai.wang@zju.edu.cn

Xiang Zhang
University of North Carolina - Charlotte
xiang.zhang@uncc.edu

Abstract

Contrastive representation learning is crucial in medical time series analysis as it alleviates dependency on labor-intensive, domain-specific, and scarce expert annotations. However, existing contrastive learning methods primarily focus on one single data level, which fails to fully exploit the intricate nature of medical time series. To address this issue, we present COMET, an innovative hierarchical framework that leverages data consistencies at all inherent levels in medical time series. Our meticulously designed model systematically captures data consistency from four potential levels: observation, sample, trial, and patient levels. By developing contrastive loss at multiple levels, we can learn effective representations that preserve comprehensive data consistency, maximizing information utilization in a self-supervised manner. We conduct experiments in the challenging patient-independent setting. We compare COMET against six baselines using three diverse datasets, which include ECG signals for myocardial infarction and EEG signals for Alzheimer's and Parkinson's diseases. The results demonstrate that COMET consistently outperforms all baselines, particularly in setup with 10% and 1% labeled data fractions across all datasets. These results underscore the significant impact of our framework in advancing contrastive representation learning techniques for medical time series. The source code is available at https://github.com/DL4mHealth/COMET.

1 Introduction

Time series data is crucial in various real-world applications, ranging from finance [1, 2, 3], engineering [4, 5], to healthcare [6, 7]. Unlike domains such as computer vision [8, 9] and natural language processing [10, 11], where human recognizable features exist, time series data often lacks readily discernible patterns, making data labeling challenging. Consequently, the scarcity of labeled data poses a significant hurdle in effectively utilizing time series for analysis and classification tasks.

To address the paucity of labeled data in time series analysis, self-supervised contrastive learning has emerged as a promising approach. For example, TimeCLR [12] proposes a DTW data augmentation for time series data; TS2vec [13] designs a cropping and masking mechanism to form positive pairs; ExpCLR [14] introduces a novel loss function to utilize continuous expert features. By leveraging the inherent consistency within unlabeled data, contrastive learning algorithms enable the extraction of effective representations without relying on explicit labels. This paradigm shift opens up possibilities for overcoming the data scarcity issue and enhancing the capabilities of time series analysis.

*These authors contributed equally to this work.

37th Conference on Neural Information Processing Systems (NeurIPS 2023).

Despite recent advancements in contrastive learning methods for time series, existing approaches fail to exploit the full potential of medical time series data, such as electroencephalogram (EEG) signals. Unlike conventional time series data, medical time series often exhibit more data levels (Figure 1), including patient, trial, sample, and observation levels. Current contrastive learning techniques exclusively employ subsets of these levels (as illustrated in Table 1). Additionally, many of these methods are tailored to specific data types, which restricts their capacity to capture the rich complexity of medical time series. For example, CLOCS [15] presents a contrastive learning method for ECG using sample and patient levels. Mixing-up [16] captures sample-level consistency through a mixing data augmentation scheme. TNC [17] exploits trial-level consistency by contrasting neighbor samples in the same trial as positive pairs. Neither of them leverages all the levels exhibited in the medical time series.

After reviewing existing contrastive learning methods within the time series domain, we consistently posed a pivotal question to ourselves: Can we design a straightforward yet appliable contrastive learning framework that can be adapted to all forms of medical time series data, akin to the classical model SimCLR [18] in the domain of contrastive learning? Our objective is to craft an innovative framework that utilizes all information within medical time series in the context of self-supervised contrastive learning. It enables us to harness patient and trial information to learn consistency across instances while leveraging the sample and observation levels' information to facilitate conventional instance discrimination.

In this paper, we propose a hierarchical framework, COMET, that systematically leverages all four levels of medical time series, namely patient, trial, sample, and observation, to reduce the reliance on labeled data. By incorporating self-supervised contrastive learning, our method aims to bridge the gap between the limited availability of labeled data and the need for robust and generalizable models in medical time series analysis. We conduct extensive experiments with six baselines on three diverse datasets in a challenging patient-independent setting. COMET outperforms SOTAs by 14% and 13% F1 score with label fractions of 10% and 1%, respectively, on EEG-based Alzheimer's detection. Further, COMET outperforms SOTAs by 0.17% and 2.66% F1 score with label fractions of 10% and 1%, respectively, on detecting Myocardial infarction with ECG. Finally, COMET outperforms SOTAs by 2% and 8% F1 score with label fractions of 10% and 1%, respectively, in the EEG-based diagnosis of Parkinson's disease. The results of downstream tasks demonstrate the effectiveness and stability of our method.

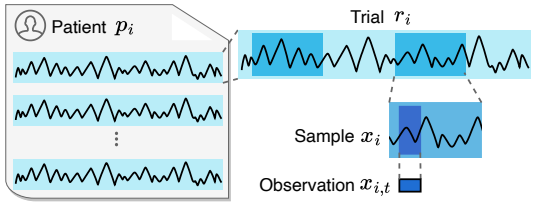


Figure 1: Structure of medical time series. Medical time series commonly have four levels (coarse to fine): patient, trial, sample, and observation. An observation is a single value in univariate time series and a vector in multivariate time series.

2 Related Work

Medical time series. Medical time series [19, 20, 21, 22] is a distinct type of time series data used for healthcare (*e.g.*, disease diagnosis, monitoring, and rehabilitation). It can be collected in a low-cost, non-invasive manner [23] or a high-cost, invasive manner [24].

Unlike general time series, which typically consist of sample and observation levels, medical time series introduces two additional data levels: patient and trial. These extra levels of data information in medical time series enable the development of specialized methods tailored to address the unique characteristics and requirements of medical time series analysis. Various types of medical time series, including EEG [28, 29, 30], ECG [31, 32], EMG [33], and EOG [34], offer valuable insights into specific medical conditions and play a crucial role in advancing healthcare practices.

Table 1: Existing methods only utilize partial levels.

Models	Patient	Trial	Sample	Observation
SimCLR [25]			✓	
TF-C [26]			✓	
Mixing-up [16]			✓	
TNC [17]		✓		
TS2vec [13]			✓	✓
TS-TCC [27]			✓	✓
CLOCS [15]	✓		✓	
COMET(Ours)	✓	✓	✓	✓

Contrastive learning for time series. Contrastive learning has demonstrated its ability to learn effective representations in various domains, including image processing [35, 18, 36], graph analysis [37, 38, 37], and time series [39]. The key idea behind contrastive representation learning is to mine data consistency by bringing similar data closer together and pushing dissimilar data further apart.

Many existing works on contrastive learning focus on general time series, while some are designed specifically for medical data [40]. One classic framework, SimCLR, transforms a single sample into two augmented views and performs contrastive learning [25]. Other settings, such as using sub-series and overlapping-series, leverage sample-level consistency [41]. TF-C [26] contrasts representations learned from the time and frequency domains to exploit sample-level consistency. Mixing-up [16] learn sample-level consistency by utilizing a mixing component as a data augmentation scheme. TS-TCC and TS2vec [27, 13] apply data augmentation at the sample-level and perform contrastive learning at the observation-level. TNC [17] learns trial-level consistency by contrasting neighboring and non-neighboring samples. NCL [42] can also be used to learn trial-level consistency if we define samples from a trial as a neighborhood. CLOCS [15] learns patient-level consistency in cardiac signal features by contrasting different leads over time.

Certain prior methods have implicitly utilized hierarchical structure [13, 15]. However, as shown in Table 1, none of these methods leverage all the levels present in the medical time series, potentially resulting in the loss of useful information during training. We explicitly present a hierarchical framework in the context of contrastive learning, which can be applied across diverse types of medical time series data. In our paper, we aim to leverage data consistency at all levels in medical time series. Our work plays a role in summarizing, inspiring, and guiding future works in self-supervised contrastive learning on medical time series.

3 Preliminaries and Problem Formulation

3.1 Medical Time Series

In this section, we clarify the key conceptions of **observation** (or measurement), **sample** (or segment), **trial** (or recording), and **patient** (or subject) in the context of medical time series (Figure 1). For better understanding, we illustrate the concepts with an example of Electroencephalography (EEG) signals for Alzheimer's Disease diagnosis (details in Appendix A).

Definition 1: Observation. *An observation $\boldsymbol{x}_{i,t} \in \mathbb{R}^F$ in medical time series data represents a single data point or a vector captured at a specific timestamp t.* Here we use i to denote the sample index (see **Definition 2**) and t to denote the timestamp. It may record physiological status, laboratory test results, vital signs, or other measurable health indicators. The observation is a single real value for univariate time series while vector for multivariate time series. The F is the feature dimension if it is a multivariate time series.

Definition 2: Sample. *A sample $\boldsymbol{x}_i = \{\boldsymbol{x}_{i,t} | t = 1, \cdots, T\}$ is a sequence of consecutive observations*, typically measured at regular intervals over a specified period (T timestamps). It can also be called a *segment* or *window*. Here we use i to denote the sample index. In the medical time series, a sample might consist of a sequence of heart rate measurements or blood pressure readings.

Definition 3: Trial. *A trial $\boldsymbol{r}_i$ is a collection of consecutive samples.* It can also be called a *record*. Here we use i to denote the trial ID. In medical time series, a trial is a continuous set of observations collected over a not-short period (*e.g.*, 30 minutes). Therefore, a trial is generally too long (*e.g.*, hundreds of thousands of observations) to feed into deep learning models for representation learning directly and is usually split into shorter subsequences (*i.e.*, samples/segments). To represent the aggregate of samples stemming from a particular trial $\boldsymbol{r}_i$ with trial ID i, we employ the notation $\mathcal{R}_i$.

Definition 4: Patient. *A patient $\boldsymbol{p}_i$ represents a collection of multiple trials stemming from a single patient.* It can also be called a *subject*. Here we use i to denote the patient ID. It is important to note that trials for a given patient may exhibit variations due to differing data collection timeframes, sensor placements, patient conditions, and other contributing factors. As shown in **Definition 3**, a trial is typically divided into many samples for better representation learning. In practical scenarios, a patient, which constitutes a cluster of trials, is also divided into samples that may share identical or distinct trial IDs but maintain the same patient ID. To represent the aggregate of samples stemming from a particular patient $\boldsymbol{p}_i$ with the corresponding patient ID i, we employ the notation $\mathcal{P}_i$.

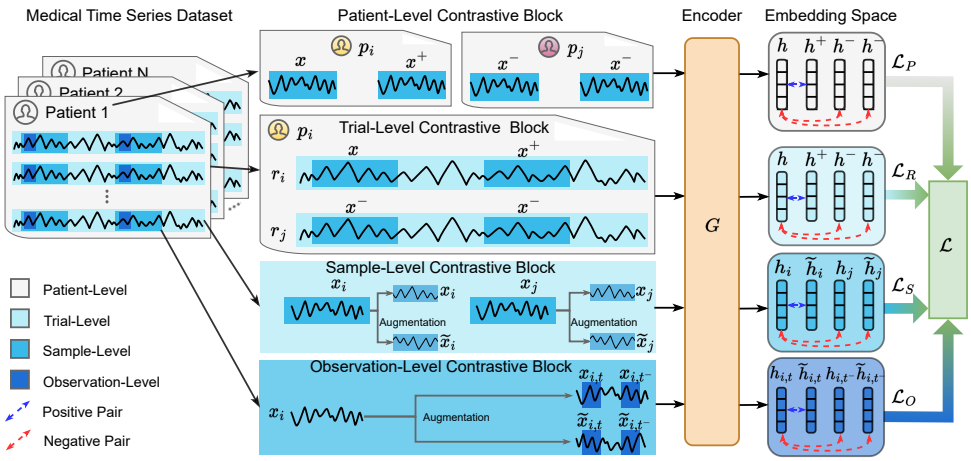


Figure 2: Overview of COMET approach. Our COMET model consists of four contrastive blocks, each illustrating the formulation of positive pairs and negative pairs at different data levels. In the observation-level contrastive, an observation $\boldsymbol{x}_{i,t}$ and its augmented view $\widetilde{\boldsymbol{x}}_{i,t}$ serve as a positive pair. Similarly, in the sample-level contrastive, a sample $\boldsymbol{x}_i$ and its augmented view $\widetilde{\boldsymbol{x}}_i$ form a positive pair. Moving to the trial-level contrastive, two samples $\boldsymbol{x}$ and $\boldsymbol{x}^+$ from the same trial $\boldsymbol{r}_i$ are considered to be a positive pair. The patient-level contrastive follows a similar pattern, where two samples $\boldsymbol{x}$ and $\boldsymbol{x}^+$ from the same patient $\boldsymbol{p}_i$ are regarded as a positive pair. Positive and corresponding negative pairs will be utilized to build contrastive loss in embedding space after being processed by encoder G.

In this work, we propose a novel hierarchical contrastive framework to learn representative and generalizable embeddings by comprehensively exploring instance discrimination at observation and sample levels and harnessing consistency across instances at trial and patient levels. Although we elaborate the proposed COMET in the context of medical time series, we note our model can possibly be extended to other time series beyond healthcare as long as extra information is available. For example, a climate dataset contains multiple meteorological satellites, each satellite contains multiple measuring units, and each unit contains multiple sensors, and every sensor can measure a specific observation at a certain timestamp. The key is to utilize all available information, excluding label data, for contrastive pre-training, such as patient ID. To adapt our approach to other domains, researchers must consider a crucial question: Does the dataset have additional information beyond sample labels? If affirmative, can this information be harnessed for contrastive learning? The example of satellite sensor application underscores the potential existence of supplementary information even in non-medical domains.

3.2 Problem Formulation

Problem (Self-Supervised Representation Learning For Medical Time Series). *Let an unlabeled dataset $\mathcal{D}$ consist of a set of patients, where each patient $\boldsymbol{p}_i$ has multiple trials, each trial $\boldsymbol{r}_i$ can be segmented into many samples, and each sample $\boldsymbol{x}_i$ comprises a series of observations. We aim to pre-train an encoder G that exploits data consistency at all available levels in a self-supervised contrastive manner. For a given time series sample $\boldsymbol{x}_i \in \mathbb{R}^{T \times F}$ with T timestamps and F feature dimensions, the encoder G learns a sample-level representation $\boldsymbol{h}_i \in \mathbb{R}^{T \times K}$, where $\boldsymbol{h}_{i,t} \in \mathbb{R}^K$ is the observation-level representation at timestamp t with K dimensions.*

By exploiting hierarchical consistency at multiple data levels, we aim to learn a representation $\boldsymbol{h}_i$ that is both representative (yielding good performance in downstream tasks) and generalizable (maintaining stability across different patients). Depending on the fine-tuning settings [18], a specific fraction of labels y_i corresponding to samples $\boldsymbol{x}_i$ are necessary.

4 Method

In this section, we first present our assumption of data consistency behind designing a hierarchical contrastive framework. Then, we describe the architecture of the proposed model COMET (Figure 2).

4.1 Hierarchical Data Consistency

Capturing data consistency is crucial in the development of a contrastive learning framework [13]. Data consistency refers to the shared commonalities preserved within the data, which provide a supervisory signal to guide model optimization. Contrastive learning captures data consistency by contrasting positive and negative data pairs, where positive pairs share commonalities and negative pairs do not. We propose consistency across four data levels: observation, sample, trial, and patient, from fine-grained to coarse-grained in the medical time series. Although we present four levels here, our model can easily be adapted to accommodate specific datasets by adding or removing data levels.

Observation-level data consistency. We assume a slightly augmented observation (*e.g.*, channel masked) will carry similar information as the original observation [13]. We use $\boldsymbol{x}_{i,t}$ as the anchor observation at timestamp t, and $\boldsymbol{x}_{i,t^-}$ as the observation at another timestamp t^- in the sample $\boldsymbol{x}_i$. We consider the anchor observation $\boldsymbol{x}_{i,t}$ and an augmented observation $\widetilde{\boldsymbol{x}}_{i,t}$ as positive pairs $(\boldsymbol{x}_{i,t}, \widetilde{\boldsymbol{x}}_{i,t})$ (with closer embeddings). Conversely, we consider the original observation $\boldsymbol{x}_{i,t}$ and the observations $\widetilde{\boldsymbol{x}}_{i,t^-}$ and $\boldsymbol{x}_{i,t^-}$ at another timestamp t^- as negative pairs $(\boldsymbol{x}_{i,t}, \widetilde{\boldsymbol{x}}_{i,t^-}), (\boldsymbol{x}_{i,t}, \boldsymbol{x}_{i,t^-})$, with distant embeddings.

Sample-level data consistency. The sample-level consistency is based on our assumption that a slightly perturbed sample (e.g., temporally masked) should carry similar information as the original sample [18, 16, 26]. We consider the anchor sample $\boldsymbol{x}_i$ and its augmented view $\widetilde{\boldsymbol{x}}_i$ as positive pair $(\boldsymbol{x}_i, \widetilde{\boldsymbol{x}}_i)$. We regard the anchor sample $\boldsymbol{x}_i$ and a different sample $\boldsymbol{x}_j$ and its augmented view $\widetilde{\boldsymbol{x}}_j$ as negative pairs: $(\boldsymbol{x}_i, \widetilde{\boldsymbol{x}}_j)$ and $(\boldsymbol{x}_i, \boldsymbol{x}_j)$.

Trial-level data consistency. We assume that samples sliced from the same trial should carry similar information compared to those obtained from different trials. For simplicity, we use $\boldsymbol{x}$ to denote the anchor sample and $\boldsymbol{x}^+$ to denote a sample from the same trial $\boldsymbol{r}_i$ as the anchor sample, while $\boldsymbol{x}^-$ to denote a sample from another trial $\boldsymbol{r}_j$. In other words, we have $\{\boldsymbol{x}, \boldsymbol{x}^+\} \in \mathcal{R}_i$ and $\boldsymbol{x}^- \in \mathcal{R}_j$. We treat sample $\boldsymbol{x}$ and the sample $\boldsymbol{x}^+$ from the same trial as positive pair $(\boldsymbol{x}, \boldsymbol{x}^+)$. We regard sample $\boldsymbol{x}$ and the sample $\boldsymbol{x}^-$ from different trials as negative pair $(\boldsymbol{x}, \boldsymbol{x}^-)$.

Patient-level data consistency. We assume samples originating from the same patient are likely to contain similar information when compared to those from different patients [15]. Here, we use $\boldsymbol{x}$ to denote the anchor sample and $\boldsymbol{x}^+$ to denote a sample from the same patient $\boldsymbol{p}_i$, while $\boldsymbol{x}^-$ from another patient $\boldsymbol{p}_j$. In other words, there are $\{\boldsymbol{x}, \boldsymbol{x}^+\} \in \mathcal{P}_i$ and $\boldsymbol{x}^- \in \mathcal{P}_j$. We have positive pair $(\boldsymbol{x}, \boldsymbol{x}^+)$ including samples from the same patient and negative pair $(\boldsymbol{x}, \boldsymbol{x}^-)$ that from different patients.

Disease-level data consistency. For completeness, we introduce disease-level data consistency, which suggests that samples associated with the same type of disease should exhibit shared patterns, even when collected from different patients in different ways. However, capturing disease-level consistency requires ground truth labels, which are not available in a self-supervised approach. As a result, we do NOT employ disease-level consistency in this paper. Nevertheless, it can be adapted for semi-supervised or supervised contrastive learning and may prove beneficial in learning domain-adaptable representations for certain diseases across patients and even datasets.

A common principle underlying all definitions is that *the X-level data consistency refers to the positive pair belonging to the same X, where X could be observation, sample, trial, patient, or disease.* We assume that *each patient is associated with only one label*, such as suffering from a specific disease, which implies that all samples from the same patient essentially originate from the same distribution. However, in cases where data from a patient is derived from multiple distributions (e.g., a patient could perform various daily activities; associated with multiple labels), the assumptions of trial-level and patient-level consistency are not satisfied. Therefore, the user can switch on the observation-level and sample-level consistency.

Building upon the concepts of data consistency, we introduce four contrastive blocks corresponding to the four data levels. Our model is highly flexible, allowing users to enable or disable any of the blocks based on the requirements of a specific task or dataset.

4.2 Observation-Level Contrastive Block

For a given time series sample $\boldsymbol{x}_i$, we apply data augmentation (such as masking) to generate an augmented sample $\widetilde{\boldsymbol{x}}_i$ [25, 43]. We input the original sample $\boldsymbol{x}_i$ and its augmented view $\widetilde{\boldsymbol{x}}_i$ into

contrastive encoder G to obtain their respective representations $\boldsymbol{h}_i = G(\boldsymbol{x}_i)$ and $\widetilde{\boldsymbol{h}}_i = G(\widetilde{\boldsymbol{x}}_i)$. It is important to note that we apply data augmentation to the samples, which indirectly extends to augmenting the observations, simplifying the encoding process. To capture observation-level consistency, we assume that, after being processed by encoder G, the representation of observation $\boldsymbol{x}_{i,t}$ is close to the representation of the augmented observation $\widetilde{\boldsymbol{x}}_{i,t}$. In contrast, it should be distant from the representations of observations $\boldsymbol{x}_{i,t^-}$ and $\widetilde{\boldsymbol{x}}_{i,t^-}$ originating from any other timestamp t^-. Specifically, our positive pair is $(\boldsymbol{x}_{i,t}, \widetilde{\boldsymbol{x}}_{i,t})$ and negative pairs are $(\boldsymbol{x}_{i,t}, \boldsymbol{x}_{i,t^-})$ and $(\boldsymbol{x}_{i,t}, \widetilde{\boldsymbol{x}}_{i,t^-})$.

Observation-level contrastive loss. The observation-level contrastive loss $\mathcal{L}_{\text{O}}$ [13] for the input sample $\boldsymbol{x}_i$ is defined as:

$$\mathcal{L}_{\text{O}} = \mathbb{E}_{\boldsymbol{x}_i \in \mathcal{D}} \left[\mathbb{E}_{t \in \mathcal{T}} \left[-\log \frac{\exp(\boldsymbol{h}_{i,t} \cdot \widetilde{\boldsymbol{h}}_{i,t})}{\sum_{t^- \in \mathcal{T}} (\exp(\boldsymbol{h}_{i,t} \cdot \widetilde{\boldsymbol{h}}_{i,t^-}) + \mathbb{1}_{[t \neq t^-]} \exp(\boldsymbol{h}_{i,t} \cdot \boldsymbol{h}_{i,t^-}))} \right] \right] \tag{1}$$

where $\mathcal{T} = \{1, \cdots, T\}$ is the set of all timestamps in sample $\boldsymbol{x}_i$ and $\cdot$ denotes dot product. The $\mathbb{1}_{[t \neq t^-]}$ is an indicator function that equals to 0 when $t = t^-$ and 1 otherwise.

4.3 Sample-Level Contrastive Block

For an input time series sample $\boldsymbol{x}_i$ and its augmented view $\widetilde{\boldsymbol{x}}_i$, we calculate their representations through $\boldsymbol{h}_i = G(\boldsymbol{x}_i)$ and $\widetilde{\boldsymbol{h}}_i = G(\widetilde{\boldsymbol{x}}_i)$. The augmentation applied here could be the same as or different from the augmentation used in Section 4.2. We assume that after passing through the encoder G, the representation of the sample $\boldsymbol{x}_i$ is close to the representation of its augmented view $\widetilde{\boldsymbol{x}}_i$, while far away from the representations of any other samples $\boldsymbol{x}_j$ and $\widetilde{\boldsymbol{x}}_j$. In specific, our positive pair is $(\boldsymbol{x}_i, \widetilde{\boldsymbol{x}}_i)$, and negative pairs are $(\boldsymbol{x}_i, \widetilde{\boldsymbol{x}}_j)$ and $(\boldsymbol{x}_i, \boldsymbol{x}_j)$.

Sample-level contrastive loss. The sample-level contrastive loss $\mathcal{L}_{\text{S}}$ [18, 43] for the input sample $\boldsymbol{x}_i$ is defined as:

$$\mathcal{L}_{\text{S}} = \mathbb{E}_{\boldsymbol{x}_i \in \mathcal{D}} \left[-\log \frac{\exp(\boldsymbol{h}_i \cdot \widetilde{\boldsymbol{h}}_i)}{\sum_{j=1}^{|\mathcal{D}|} (\exp(\boldsymbol{h}_i \cdot \widetilde{\boldsymbol{h}}_j) + \mathbb{1}_{[i \neq j]} \exp(\boldsymbol{h}_i \cdot \boldsymbol{h}_j))} \right] \tag{2}$$

where $|\mathcal{D}|$ represents the total number of samples in the dataset $\mathcal{D}$ and $\cdot$ denotes dot product. The $\mathbb{1}_{[i \neq j]}$ is an indicator function that equals 0 when $i = j$ and 1 otherwise.

4.4 Trial-Level Contrastive Block

For an input sample $\boldsymbol{x} \in \mathcal{R}_i$, where $\mathcal{R}_i$ is a collection of all samples segmented from trial $\boldsymbol{r}_i$, we feed it into the contrastive encoder G to generate a sample-level representation $\boldsymbol{h} = G(\boldsymbol{x})$. To seize trial-level data consistency, we assume that the representation of the anchor sample $\boldsymbol{x} \in \mathcal{R}_i$ is close to the representation of sample $\boldsymbol{x}^+$ that also come from the trial $\boldsymbol{r}_i$. In contrast, the representation of the anchor sample $\boldsymbol{x}$ is far away from the representation of sample $\boldsymbol{x}^-$ that come from a different trial $\boldsymbol{r}_j$, where $\boldsymbol{x}^- \in \mathcal{R}_j$. In other words, we have positive pair $(\boldsymbol{x}, \boldsymbol{x}^+)$ and negative pair $(\boldsymbol{x}, \boldsymbol{x}^-)$.

Trial-level contrastive loss. The trial-level contrastive loss $\mathcal{L}_{\text{R}}$ [15, 18] for the input sample $\boldsymbol{x}$ is defined as:

$$\mathcal{L}_{\text{R}} = \mathbb{E}_{\boldsymbol{x} \in \mathcal{D}} \left[\mathbb{E}_{\boldsymbol{x}^+ \in \mathcal{R}_i} \left[-\log \frac{\exp(\text{sim}(\boldsymbol{h}, G(\boldsymbol{x}^+))/\tau)}{\sum_{j=1}^{J} \sum_{\boldsymbol{x}^- \in \mathcal{R}_j} (\exp(\text{sim}(\boldsymbol{h}, G(\boldsymbol{x}^-))/\tau))} \right] \right] \tag{3}$$

where J is the total number of trials in the dataset $\mathcal{D}$. The $\text{sim}(\boldsymbol{u}, \boldsymbol{v}) = \boldsymbol{u}^T \boldsymbol{v} / \|\boldsymbol{u}\| \|\boldsymbol{v}\|$ denotes the cosine similarity, and τ is a temperature parameter to adjust the scale. The $G(\boldsymbol{x}^+)$ and $G(\boldsymbol{x}^-)$ are learned representations of samples $\boldsymbol{x}^+ \in \mathcal{R}_i$ and $\boldsymbol{x}^- \in \mathcal{R}_j$, respectively. To measure the trial-level loss for sample $\boldsymbol{x}$, we iterate all the $\boldsymbol{x}^+$ in $\mathcal{R}_i$, and averaging across $|\mathcal{R}_i| - 1$ positive pairs.

In this block, we do NOT learn a trial-level embedding representing the entire trial. Instead, we learn a representation for each sample within the trial while considering trial-level data consistency. Similarly, we follow this protocol for the patient-level contrastive block.

4.5 Patient-Level Contrastive Block

For an input sample $\boldsymbol{x} \in \mathcal{P}_i$, where $\mathcal{P}_i$ denotes all samples from patient $\boldsymbol{p}_i$, we feed it into the contrastive encoder G to generate a sample-level representation $\boldsymbol{h} = G(\boldsymbol{x})$. Similar to the above

trial-level contrastive block, we have positive pair $(\boldsymbol{x}, \boldsymbol{x}^+)$ and negative pair $(\boldsymbol{x}, \boldsymbol{x}^-)$, in which $\boldsymbol{x}^+$ come from the same patient while $\boldsymbol{x}^-$ come from a different patient.

Patient-level contrastive loss. The patient-level contrastive loss $\mathcal{L}_{\text{P}}$ [15] for the input sample $\boldsymbol{x}$ is defined as:

$$\mathcal{L}_{\text{P}} = \mathbb{E}_{\boldsymbol{x}\in\mathcal{D}}\left[\mathbb{E}_{\boldsymbol{x}^+\in\mathcal{P}_i}\left[-\log\frac{\exp(\text{sim}(\boldsymbol{h}, G(\boldsymbol{x}^+))/\tau)}{\sum_{j=1}^{M}\sum_{\boldsymbol{x}^-\in\mathcal{P}_j}(\exp(\text{sim}(\boldsymbol{h}, G(\boldsymbol{x}^-))/\tau))}\right]\right] \tag{4}$$

where M is the total number of patients in the dataset $\mathcal{D}$. In this block, the $G(\boldsymbol{x}^+)$ and $G(\boldsymbol{x}^-)$ are learned representations of samples $\boldsymbol{x}^+ \in \mathcal{P}_i$ and $\boldsymbol{x}^- \in \mathcal{P}_j$, respectively.

4.6 Overall Loss Function

The overall loss function $\mathcal{L}$ consists of four loss terms The observation-level loss $\mathcal{L}_{\text{O}}$ and sample-level loss $\mathcal{L}_{\text{S}}$ encourage the encoder to learn robust representations that are invariant to perturbations. The trial-level loss $\mathcal{L}_{\text{R}}$ and patient-level loss $\mathcal{L}_{\text{P}}$ compel the encoder to learn cross-sample features within a trial or a patient. In summary, the overall loss function of the proposed COMET model is:

$$\mathcal{L} = \lambda_1\mathcal{L}_{\text{O}} + \lambda_2\mathcal{L}_{\text{S}} + \lambda_3\mathcal{L}_{\text{R}} + \lambda_4\mathcal{L}_{\text{P}} \tag{5}$$

where $\lambda_1, \lambda_2, \lambda_3, \lambda_4 \in [0, 1]$ are hyper-coefficients that control the relative importance and adjust the scales of each level's loss. Users can simply turn off specific data levels by setting λ of those levels to 0. We set $\lambda_1 + \lambda_2 + \lambda_3 + \lambda_4 = 1$. We calculate the total loss by taking the expectation of $\mathcal{L}$ across all samples $\boldsymbol{x} \in \mathcal{D}$. In practice, the contrastive losses are calculated within a mini-batch.

5 Experiments

We compare the COMET model with six baselines on three datasets. Following the setup in SimCLR [18], we use unlabeled data to pre-train encoder G and evaluate it in two downstream settings (Appendix E): partial fine-tuning (P-FT; *i.e.*, linear evaluation [18]) and full fine-tuning (F-FT). All datasets are split into training, validation, and test sets in **patient-independent** setting (Figure 3), which is very challenging due to patient variations (More explanations in Appendix F.1).

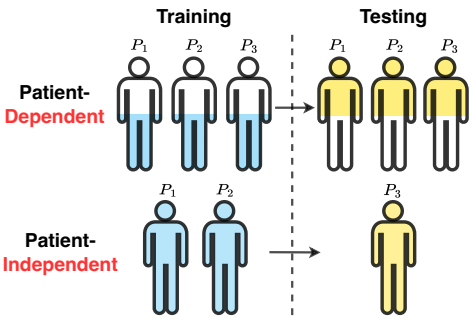


Figure 3: Patient-dependent/independent Setting. In the patient-dependent setting, samples from the same patient can appear in both the training and test sets. In contrast, in the patient-independent setting, samples from the same patient are exclusively included in the training or test set.

Datasets. (1) **AD** [44] has 23 patients, 663 trials, and 5967 multivariate EEG samples. There are 4329, 891, and 747 samples in training, validation, and test sets. The sampling rate (frequency) is 256Hz. Each sample is a one-second interval with 256 observations. A binary label based on whether the patient has Alzheimer's disease is assigned to each sample. (2) **PTB** [45] has 198 patients, 6237 trials, and 62370 multivariate ECG samples. There are 53950, 3400, and 5020 samples in training, validation, and test sets. The sampling rate (frequency) is 250Hz. Each sample is a heartbeat with 300 observations. A binary label based on whether the patient has Myocardial infarction is assigned to each sample. (3) **TDBRAIN** [46] has 72 patients, 624 trials, and 11856 multivariate EEG samples. There are 8208, 1824, and 1824 samples in training, validation, and test sets. The sampling rate (frequency) is 256Hz. Each sample is a one-second interval with 256 observations. A binary label based on whether the patient has Parkinson's disease is assigned to each sample. See appendix D for more details about data statistics, train-test split, and data preprocessing.

Baselines. We compare with 6 state-of-the-art methods: TS2vec [13], Mixing-up [16], TS-TCC [27], SimCLR [43], CLOCS [15] and TF-C [26]. Since TF-C is designed for transfer learning, we implement its downstream tasks the same as ours for a fair comparison. The evaluation metrics are accuracy, precision (macro-averaged), recall(macro-averaged), F1 score(macro-averaged, AUROC(macro-averaged), and AUPRC(macro-averaged).

Table 2: Partial fine-tuning results. A logistic regression(LG) classifier L is added on top of a frozen encoder G on dataset AD. Only fine-tuning the classifier L.

Models	Accuracy	Precision	Recall	F1 score	AUROC	AUPRC
TS2vec	66.48±5.33	67.72±5.09	67.40±5.20	66.32±5.46	74.12±6.88	72.96±7.21
TF-C	77.03±2.80	75.79±5.07	64.27±5.03	64.85±5.56	80.71±4.03	79.27±4.15
Mixing-up	46.16±1.38	52.62±4.90	50.81±1.32	37.37±1.98	64.42±6.49	62.85±6.07
TS-TCC	59.71±8.63	61.66±8.63	60.33±8.26	58.66±8.39	67.53±10.04	68.33±9.37
SimCLR	57.16±2.05	56.67±3.91	53.57±2.21	49.11±4.26	56.67±3.91	52.10±1.41
CLOCS	66.99±2.84	67.17±2.96	67.33±2.99	66.91±2.88	73.71±3.62	72.58±3.54
COMET (Ours)	**76.09**±4.21	**77.36**±3.97	**74.68**±4.62	**74.80**±4.83	**81.30**±4.97	**80.50**±5.31

Implementation. We use a ten residual blocks dilated CNN module [47] as backbones for encoders G. To preserve the magnitude feature of time series, which is often an important feature in time series, we utilize timestamp masking as proposed in [13] as an augmentation method. While many existing works apply data augmentation directly on raw time series, we use a fully connected projection layer to map the raw input into an embedding with a higher feature dimension. This strategy helps prevent situations where some parts of the raw data are zero, which could result in ineffective augmentation if timestamp masking were directly applied. Additionally, to ensure stability during training, we incorporate a hierarchical loss [13], which encapsulates the observation and sample-level losses. To guarantee there are samples with the same trial and patient IDs in a batch for trial and patient levels contrasting, we design two specific shuffle functions. See Appendix C for further details. For datasets whose trial level is absent or limited to one trial per patient, we provide a solution in Appendix F.2.

We conduct experiments with five seeds(41-45) based on the same data split to account for variance and report the mean and standard deviation. All experiments expect baseline SimCLR run on NVIDIA RTX 4090. Baseline SimCLR runs on Google Colab NVIDIA A100. See further implementation details in Appendix E.

5.1 Results on Partial Fine-tuning

Setup. Linear evaluation is a widely used P-FT setup to evaluate the quality of representation, where a linear classifier is trained on top of a frozen network for sample classification [18, 15, 48, 49]. This evaluation method allows for a quick assessment of representation quality. In linear evaluation, we add a logistic regression classifier L on top of the pre-trained encoder G. The training process utilizes 100% labeled training data for sample classification. Notably, the parameters of the encoder G are frozen during training, ensuring that only the classifier L is fine-tuned. See further details in E.1.

Results. We report the experimental results of the P-FT setup on AD in Table 2. On average, our COMET model claims a large margin of more than 7% over all baselines on the F1 score on AD.

5.2 Results on Full Fine-tuning

Setup. In an F-FT setup, we add a two-layer fully connected network classifier P to the pre-trained encoder G. The training process utilizes 100%, 10%, and 1% of the labeled training data for sample classification, respectively. Unlike the P-FT approach, both the encoder G and classifier P are trainable now, allowing for fine-tuning of the entire network structure. See further details in E.2.

Results. The results for F-FT on the three datasets are presented in Table 3 and Table 4. In general, COMET demonstrates success in 46 out of 54 tests conducted across the three datasets, considering six different evaluation metrics. With 100% labeled data, the F1 score of COMET outperforms the best baseline, TS2Vec, by 2% on the AD dataset and surpasses the best baseline, Mixing-up, by 4% on the TDBrain dataset. Furthermore, it achieves a result comparable to the best baseline, TF-C, on the PTB dataset.

Notably, the F1 score of COMET surpasses the best baseline, TF-C, by 14% and 13% with label fractions of 10% and 1%, respectively, on the AD dataset. Additionally, on the TDBrain dataset, the F1 score of COMET outperforms the best baseline, Mixing-up, by 2% and 8% with label fractions of 10% and 1%, respectively. Similarly, the F1 score of COMET outperforms the best baseline, CLOCS, by 0.17% and 2.66% with label fractions of 10% and 1%, respectively, on the PTB dataset. It is interesting to observe that the F1 score of COMET with label fractions of 10% and 1% outperforms a label fraction of 100% on the AD and PTB datasets. This suggests a potential overfitting of COMET to the training data. A similar phenomenon is observed with SimCLR, TF-C, CLOCS, and TS2vec, where a higher fraction of labeled data did not necessarily lead to improved performance.

Table 3: Full fine-tuning results of EEG datasets. A two-layer fully connected networks classifier P is trained on top of pre-trained encoder G on EEG datasets AD and TDBRAIN. Fine-tuning the whole network includes G and P. We use 100%, 10%, and 1% of labeled training data in AD and TDBRAIN.

Datasets	Fraction	Models	Accuracy	Precision	Recall	F1 score	AUROC	AUPRC
AD	100%	**TS2vec**	81.26±2.08	81.21±2.14	81.34±2.04	81.12±2.06	89.20±1.76	88.94±1.85
		TF-C	75.31±8.27	75.87±8.73	74.83±8.98	74.54±8.85	79.45±10.23	79.33±10.57
		Mixing-up	65.68±7.89	72.61±4.21	68.25±6.97	63.98±9.92	84.63±5.04	83.46±5.48
		TS-TCC	73.55±10.00	77.22±6.13	73.83±9.65	71.86±11.59	86.17±5.11	85.73±5.11
		SimCLR	54.77±1.97	50.15±7.02	50.58±1.92	43.18±4.27	50.15±7.02	50.42±1.06
		CLOCS	78.37±6.00	83.99±2.11	76.14±7.03	75.78±7.93	91.17±2.51	90.72±3.05
		COMET (Ours)	**84.50±4.46**	**88.31±2.42**	**82.95±5.39**	**83.33±5.15**	**94.44±2.37**	**94.43±2.48**
	10%	**TS2vec**	73.28±4.34	74.14±4.33	73.52±3.77	73.00±4.18	81.66±5.20	81.58±5.11
		TF-C	75.66±11.21	75.48±11.48	75.58±11.59	75.38±11.47	81.38±14.19	81.56±13.68
		Mixing-up	59.38±3.33	64.85±4.38	61.94±3.42	58.17±3.41	75.02±6.14	73.44±5.82
		TS-TCC	77.83±6.90	79.73±7.49	76.18±7.21	76.43±7.56	84.12±7.32	84.12±7.61
		SimCLR	56.09±2.25	53.81±5.74	51.73±2.59	44.10±4.84	53.81±5.74	51.08±1.53
		CLOCS	76.97±3.01	81.70±3.21	74.69±3.26	74.75±3.61	86.91±3.61	86.70±3.64
		COMET (Ours)	**91.43±3.12**	**92.52±2.36**	**90.71±3.56**	**91.14±3.31**	**96.44±2.84**	**96.48±2.82**
	1%	**TS2vec**	64.93±3.53	65.28±3.52	65.14±3.59	64.64±3.58	70.56±5.38	68.97±5.75
		TF-C	75.66±12.61	75.26±13.91	74.77±13.64	74.33±14.45	78.96±16.17	81.89±12.50
		Mixing-up	63.67±1.20	65.02±2.09	64.64±1.63	63.55±1.21	71.95±3.39	70.15±3.70
		TS-TCC	53.04±8.80	52.39±13.73	52.00±8.01	44.69±12.24	48.89±10.59	51.41±7.81
		SimCLR	55.42±2.43	52.18±5.55	51.37±2.76	45.02±4.79	52.18±5.55	50.87±1.45
		CLOCS	64.50±4.16	65.17±4.72	63.73±4.86	63.01±5.11	69.16±6.75	68.15±7.20
		COMET (Ours)	**88.22±3.36**	**88.55±2.73**	**88.56±3.14**	**88.14±3.37**	**96.05±1.36**	**96.12±1.31**
TDBrain	100%	**TS2vec**	80.21±1.69	81.38±1.97	80.21±1.69	80.07±1.69	89.57±2.31	89.60±2.37
		TF-C	66.62±1.76	67.15±1.64	66.62±1.76	66.35±1.91	65.43±6.13	66.18±4.90
		Mixing-up	81.47±1.07	82.11±1.12	81.47±1.07	81.38±1.08	90.48±0.89	90.51±0.94
		TS-TCC	77.42±2.86	77.68±2.93	77.42±2.86	77.37±2.86	87.37±3.06	87.61±3.22
		SimCLR	58.43±1.77	59.48±1.95	58.43±1.77	57.30±2.07	59.48±1.95	55.05±1.18
		CLOCS	78.16±1.13	79.49±1.61	78.16±1.13	77.91±1.12	88.49±1.95	88.83±1.94
		COMET (Ours)	**85.47±1.16**	**85.68±1.20**	**85.47±1.16**	**85.45±1.16**	**93.73±1.02**	**93.96±0.99**
	10%	**TS2vec**	72.39±1.13	74.49±1.73	72.39±1.13	71.80±1.05	80.71±1.90	80.06±1.87
		TF-C	59.14±3.04	59.34±3.19	59.14±3.04	58.98±2.94	59.56±4.10	59.65±2.99
		Mixing-up	77.50±2.07	**80.09±1.92**	77.50±2.07	76.99±2.28	87.29±1.34	87.13±1.37
		TS-TCC	71.23±1.57	78.78±0.66	71.23±1.57	69.18±2.25	80.56±1.98	80.21±2.21
		SimCLR	59.79±2.09	60.50±1.90	59.79±2.09	59.06±2.82	60.50±1.90	55.96±1.41
		CLOCS	75.04±2.65	75.97±2.86	75.04±2.65	74.83±2.66	84.25±3.27	84.37±3.52
		COMET (Ours)	**79.28±4.64**	79.83±4.83	**79.28±4.64**	**79.19±4.62**	**88.39±4.13**	**88.38±3.96**
	1%	**TS2vec**	58.16±2.30	58.47±2.48	58.16±2.30	57.83±2.19	61.26±2.51	60.95±2.48
		TF-C	53.16±2.11	53.19±2.10	53.16±2.11	52.99±2.18	52.11±2.95	59.11±1.88
		Mixing-up	63.91±2.39	64.29±2.57	63.91±2.39	63.70±2.30	69.61±3.64	68.89±4.07
		TS-TCC	49.14±1.88	51.10±8.54	49.14±1.88	40.82±4.17	48.05±4.28	50.17±3.94
		SimCLR	59.46±1.79	59.77±1.87	59.46±1.79	59.16±1.81	59.77±1.87	55.69±1.27
		CLOCS	58.85±4.93	59.03±5.00	58.85±4.93	58.38±5.45	62.57±6.38	62.14±6.34
		COMET (Ours)	**72.93±7.21**	**74.20±7.68**	**72.93±7.21**	**72.57±7.37**	**78.72±8.42**	**77.71±9.10**

As a result, COMET demonstrates its superiority and stability across all the datasets. Furthermore, COMET outperforms SOTAs methods significantly with 10% and 1% label fractions, highlighting the effectiveness of our contrastive pre-training approach in reducing the reliance on labeled data.

5.3 Ablation Study, Visualization, Additional Downstream Tasks, and Heavy Duty Baseline

Ablation study. We verify the effectiveness of each contrastive block and other COMET variants. Besides, we study the impact of hyperparameter λ. (Appendix G.1)

Visualization. We visualize the embedding space and show our model can learn more distinctive and robust representations. (Appendix G.2)

Additional downstream tasks. Apart from the classification tasks in Section 5.1-5.2, we show the proposed COMET outperforms baselines in a wide range of downstream tasks, including clustering and anomaly detection. (Appendix G.3)

Heavy duty baseline. We show the superiority of our model does NOT result from the newly added contrastive blocks (*i.e.*, increased model parameters). COMET outperforms the heavy-duty SimCLR and TS2Vec with four contrastive blocks and four times pre-training epochs. (Appendix G.4)

Table 4: Full fine-tuning results of ECG datasets. A two-layer fully connected networks classifier P is trained on top of pre-trained encoder G on ECG dataset PTB. Fine-tuning the whole network includes G and P. We use 100%, 10%, and 1% of labeled training data in PTB.

Datasets	Fraction	Models	Accuracy	Precision	Recall	F1 score	AUROC	AUPRC
PTB	100%	**TS2vec**	85.14±1.66	87.82±2.21	76.84±3.99	79.66±3.63	90.50±1.59	**90.07±1.73**
		TF-C	87.50±2.43	85.50±3.04	**82.68±4.04**	**83.77±3.50**	77.59±19.22	80.62±15.10
		Mixing-up	87.61±1.48	**89.56±2.80**	79.30±1.67	82.56±2.00	89.29±1.26	88.94±1.04
		TS-TCC	82.24±3.55	85.28±5.08	69.46±5.85	72.08±6.85	87.60±2.51	86.26±3.00
		SimCLR	84.19±1.32	85.85±1.89	73.89±2.95	76.84±2.80	85.85±1.89	70.70±2.36
		CLOCS	86.39±2.76	87.06±2.81	77.95±4.79	80.71±4.78	90.41±2.07	87.35±3.36
		COMET (Ours)	**87.84±1.98**	87.67±1.72	81.14±3.68	83.45±3.22	**92.95±1.56**	87.47±2.82
	10%	**TS2vec**	82.49±4.71	80.39±5.04	**83.35±0.91**	80.18±4.04	93.03±1.03	**92.81±0.97**
		TF-C	85.37±1.23	82.80±2.35	79.94±0.71	81.09±1.14	81.57±15.60	84.57±8.12
		Mixing-up	87.05±1.41	86.56±3.24	80.61±2.68	82.62±1.99	89.28±1.43	87.22±2.76
		TS-TCC	83.38±1.53	85.11±2.11	72.24±2.45	75.27±2.64	86.06±1.76	84.34±2.08
		SimCLR	83.84±2.15	87.19±1.34	72.51±4.63	75.44±4.77	87.19±1.34	69.99±3.84
		CLOCS	88.25±2.48	88.64±2.12	81.40±4.64	83.84±4.03	91.91±2.40	89.76±3.94
		COMET (Ours)	**88.49±3.28**	**88.98±2.60**	81.65±6.00	**84.01±5.61**	**94.83±1.08**	92.48±2.22
	1%	**TS2vec**	81.95±1.85	78.78±0.98	83.83±0.36	79.77±1.44	90.99±1.34	88.69±1.44
		TF-C	85.82±2.34	82.79±3.20	81.86±2.36	82.21±2.62	89.14±2.89	89.47±2.92
		Mixing-up	84.71±4.11	82.33±6.07	78.17±4.89	79.81±5.33	89.04±3.57	84.67±5.16
		TS-TCC	78.61±1.19	78.27±1.79	64.56±2.53	66.23±3.16	79.97±2.17	76.98±2.00
		SimCLR	84.19±1.49	87.15±1.27	73.21±3.43	76.31±3.26	87.15±1.27	70.58±2.71
		CLOCS	88.80±2.82	88.11±4.37	84.47±3.41	85.57±3.41	94.73±1.59	**94.04±2.14**
		COMET (Ours)	**90.52±1.90**	**88.58±2.93**	**88.23±1.98**	**88.23±2.10**	**95.08±1.50**	93.78±1.98

6 Conclusion

This paper introduces COMET, a hierarchical contrastive representation learning framework tailored for medical time series. COMET leverages all data levels of medical time series, including patient, trial, sample, and observation levels, to capture the intricate complexities of the data. Through extensive experiments on three diverse datasets, we demonstrate that our method surpasses existing state-of-the-art methods in medical time series analysis. Our framework also shows its effectiveness in patient-independent downstream tasks, highlighting its potential to advance medical time series analysis and improve patient care and diagnosis.

One limitation of our work is the presence of label conflicts between different levels. In patient-level consistency, we assume all samples belonging to the same patient are positive while others are negative. In trial-level consistency, we consider samples from the same trial as positive samples and others as negative. This means a positive pair at the patient level may be considered negative at the trial level, as they do not belong to the same trial.

In future research, we aim to investigate the efficacy of our method across a wider range of datasets, with a particular focus on ECG datasets. Additionally, we intend to explore approaches to integrate disease-level consistency within our self-supervised contrastive framework.

We discuss the **broader impacts** with potential negative social impacts in Appendix H.

Acknowledgments and Disclosure of Funding

This work is partially supported by the National Science Foundation under Grant No. 2245894. Any opinions, findings, conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the funders.

References

[1] Dawei Cheng, Fangzhou Yang, Sheng Xiang, and Jin Liu. Financial time series forecasting with multi-modality graph neural network. Pattern Recognition, 121:108218, 2022.

[2] Yajiao Tang, Zhenyu Song, Yulin Zhu, Huaiyu Yuan, Maozhang Hou, Junkai Ji, Cheng Tang, and Jianqiang Li. A survey on machine learning models for financial time series forecasting. Neurocomputing, 512:363–380, 2022.

[3] Xiaoyu He, Suixiang Shi, Xiulin Geng, and Lingyu Xu. Information-aware attention dynamic synergetic network for multivariate time series long-term forecasting. Neurocomputing, 500:143–154, 2022.

[4] Zhiwen Chen, Jiamin Xu, Tao Peng, and Chunhua Yang. Graph convolutional network-based method for fault diagnosis using a hybrid of measurement and prior knowledge. IEEE transactions on cybernetics, 52(9):9157–9169, 2021.

[5] Quan Qian, Yi Qin, Jun Luo, and Dengyu Xiao. Cross-machine transfer fault diagnosis by ensemble weighting subdomain adaptation network. IEEE Transactions on Industrial Electronics, 2023.

[6] Catherine Arsenault, Anna Gage, Min Kyung Kim, Neena R Kapoor, Patricia Akweongo, Freddie Amponsah, Amit Aryal, Daisuke Asai, John Koku Awoonor-Williams, Wondimu Ayele, et al. Covid-19 and resilience of healthcare systems in ten countries. Nature Medicine, 28(6):1314–1324, 2022.

[7] Andrea Brizzi, Charles Whittaker, Luciana MS Servo, Iwona Hawryluk, Carlos A Prete Jr, William M de Souza, Renato S Aguiar, Leonardo JT Araujo, Leonardo S Bastos, Alexandra Blenkinsop, et al. Spatial and temporal fluctuations in covid-19 fatality rates in brazilian hospitals. Nature medicine, 28(7):1476–1485, 2022.

[8] Danyang Tu, Xiongkuo Min, Huiyu Duan, Guodong Guo, Guangtao Zhai, and Wei Shen. End-to-end human-gaze-target detection with transformers. In 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 2192–2200. IEEE, 2022.

[9] Jaime Spencer, Richard Bowden, and Simon Hadfield. Medusa: Universal feature learning via attentional multitasking. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 3800–3809, 2022.

[10] Haris Bin Zia, Ignacio Castro, Arkaitz Zubiaga, and Gareth Tyson. Improving zero-shot cross-lingual hate speech detection with pseudo-label fine-tuning of transformer language models. In Proceedings of the International AAAI Conference on Web and Social Media, volume 16, pages 1435–1439, 2022.

[11] Tuan Dinh, Yuchen Zeng, Ruisu Zhang, Ziqian Lin, Michael Gira, Shashank Rajput, Jy-yong Sohn, Dimitris Papailiopoulos, and Kangwook Lee. Lift: Language-interfaced fine-tuning for non-language machine learning tasks. Advances in Neural Information Processing Systems, 35:11763–11784, 2022.

[12] Xinyu Yang, Zhenguo Zhang, and Rongyi Cui. Timeclr: A self-supervised contrastive learning framework for univariate time series representation. Knowledge-Based Systems, 245:108606, 2022.

[13] Zhihan Yue, Yujing Wang, Juanyong Duan, Tianmeng Yang, Congrui Huang, Yunhai Tong, and Bixiong Xu. Ts2vec: Towards universal representation of time series. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 36, pages 8980–8987, 2022.

[14] Manuel T Nonnenmacher, Lukas Oldenburg, Ingo Steinwart, and David Reeb. Utilizing expert features for contrastive learning of time-series representations. In International Conference on Machine Learning, pages 16969–16989. PMLR, 2022.

[15] Dani Kiyasseh, Tingting Zhu, and David A Clifton. Clocs: Contrastive learning of cardiac signals across space, time, and patients. In International Conference on Machine Learning, pages 5606–5615. PMLR, 2021.

[16] Kristoffer Wickstrøm, Michael Kampffmeyer, Karl Øyvind Mikalsen, and Robert Jenssen. Mixing up contrastive learning: Self-supervised representation learning for time series. Pattern Recognition Letters, 155:54–61, 2022.

[17] Sana Tonekaboni, Danny Eytan, and Anna Goldenberg. Unsupervised representation learning for time series with temporal neighborhood coding. In ICLR, 2021.

[18] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In International conference on machine learning, pages 1597–1607. PMLR, 2020.

[19] Minji Lee, Leandro RD Sanz, Alice Barra, Audrey Wolff, Jaakko O Nieminen, Melanie Boly, Mario Rosanova, Silvia Casarotto, Olivier Bodart, Jitka Annen, et al. Quantifying arousal and awareness in altered states of consciousness using interpretable deep learning. Nature communications, 13(1):1064, 2022.

[20] Chenxi Sun, Hongyan Li, Moxian Song, and Shenda Hong. A ranking-based cross-entropy loss for early classification of time series. IEEE Transactions on Neural Networks and Learning Systems, 2023.

[21] Huizi Cui, Lingge Zhou, Yan Li, and Bingyi Kang. Belief entropy-of-entropy and its application in the cardiac interbeat interval time series analysis. Chaos, Solitons & Fractals, 155:111736, 2022.

[22] Zongxing Xie, Bing Zhou, Xi Cheng, Elinor Schoenfeld, and Fan Ye. Passive and context-aware in-home vital signs monitoring using co-located uwb-depth sensor fusion. ACM Transactions on Computing for Healthcare, 3(4):1–31, 2022.

[23] Martin Krause, Shannon K McWilliams, Kenneth J Bullard, Lena M Mayes, Leslie C Jameson, Susan K Mikulich-Gilbertson, Ana Fernandez-Bustamante, and Karsten Bartels. Neostigmine versus sugammadex for reversal of neuromuscular blockade and effects on re-intubation for respiratory failure or newly initiated non-invasive ventilation–an interrupted time series design. Anesthesia and analgesia, 131(1):141, 2020.

[24] Mohammad-Parsa Hosseini, Tuyen X Tran, Dario Pompili, Kost Elisevich, and Hamid Soltanian-Zadeh. Multimodal data analysis of epileptic eeg and rs-fmri via deep learning and edge computing. Artificial Intelligence in Medicine, 104:101813, 2020.

[25] Johannes Pöppelbaum, Gavneet Singh Chadha, and Andreas Schwung. Contrastive learning based self-supervised time-series analysis. Applied Soft Computing, 117:108397, 2022.

[26] Xiang Zhang, Ziyuan Zhao, Theodoros Tsiligkaridis, and Marinka Zitnik. Self-supervised contrastive pre-training for time series via time-frequency consistency. In NeurIPS, 2022.

[27] 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 IJCAI, pages 2352–2359, 2021.

[28] Siyi Tang, Jared Dunnmon, Khaled Kamal Saab, Xuan Zhang, Qianying Huang, Florian Dubost, Daniel Rubin, and Christopher Lee-Messer. Self-supervised graph neural networks for improved electroencephalographic seizure analysis. In International Conference on Learning Representations, 2021.

[29] Chaoqi Yang, M Brandon Westover, and Jimeng Sun. Manydg: Many-domain generalization for healthcare applications. In The Eleventh International Conference on Learning Representations, 2023.

[30] Xiaodong Qu, Zepeng Hu, Zhaonan Li, and Timothy J Hickey. Ensemble methods and lstm outperformed other eight machine learning classifiers in an eeg-based bci experiment. In International Conference on Learning Representations, 2020.

[31] Uri Shaham, Jonathan Svirsky, Ori Katz, and Ronen Talmon. Discovery of single independent latent variable. Advances in Neural Information Processing Systems, 35:25251–25263, 2022.

[32] Xu Chen and Brett Wujek. A unified framework for automatic distributed active learning. IEEE Transactions on Pattern Analysis & Machine Intelligence, 44(12):9774–9786, 2022.

[33] Yuanchao Dai, Jing Wu, Yuanzhao Fan, Jin Wang, Jianwei Niu, Fei Gu, and Shigen Shen. Mseva: A musculoskeletal rehabilitation evaluation system based on emg signals. ACM Transactions on Sensor Networks, 19(1):1–23, 2022.

[34] Yingying Jiao, Yini Deng, Yun Luo, and Bao-Liang Lu. Driver sleepiness detection from eeg and eog signals using gan and lstm networks. Neurocomputing, 408:100–111, 2020.

[35] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. Advances in neural information processing systems, 33:21271–21284, 2020.

[36] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. arXiv preprint arXiv:1807.03748, 2018.

[37] Chunhui Zhang, Hongfu Liu, Jundong Li, Yanfang Ye, and Chuxu Zhang. Contrastive graph few-shot learning. In NeurIPS 2022 Workshop: New Frontiers in Graph Learning, 2022.

[38] Zekun Tong, Yuxuan Liang, Henghui Ding, Yongxing Dai, Xinke Li, and Changhu Wang. Directed graph contrastive learning. Advances in Neural Information Processing Systems, 34:19580–19593, 2021.

[39] Aniruddh Raghu, Payal Chandak, Ridwan Alam, John Guttag, and Collin Stultz. Contrastive pre-training for multimodal medical time series. In NeurIPS 2022 Workshop on Learning from Time Series for Health, 2022.

[40] Ziyu Liu, Azadeh Alavi, Minyi Li, and Xiang Zhang. Self-supervised contrastive learning for medical time series: A systematic review. Sensors, 23(9):4221, 2023.

[41] Jean-Yves Franceschi, Aymeric Dieuleveut, and Martin Jaggi. Unsupervised scalable representation learning for multivariate time series. Advances in neural information processing systems, 32, 2019.

[42] Hugo Yèche, Gideon Dresdner, Francesco Locatello, Matthias Hüser, and Gunnar Rätsch. Neighborhood contrastive learning applied to online patient monitoring. In International Conference on Machine Learning, pages 11964–11974. PMLR, 2021.

[43] Chi Ian Tang, Ignacio Perez-Pozuelo, Dimitris Spathis, and Cecilia Mascolo. Exploring contrastive learning in human activity recognition for healthcare. Machine Learning for Mobile Health Workshop at NeurIPS, 2020.

[44] J Escudero, Daniel Abásolo, Roberto Hornero, Pedro Espino, and Miguel López. Analysis of electroencephalograms in alzheimer's disease patients with multiscale entropy. Physiological measurement, 27(11):1091, 2006.

[45] PhysioToolkit PhysioBank. Physionet: components of a new research resource for complex physiologic signals. Circulation, 101(23):e215–e220, 2000.

[46] Hanneke van Dijk, Guido van Wingen, Damiaan Denys, Sebastian Olbrich, Rosalinde van Ruth, and Martijn Arns. The two decades brainclinics research archive for insights in neurophysiology (tdbrain) database. Scientific data, 9(1):333, 2022.

[47] Shaojie Bai, J Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. arXiv preprint arXiv:1803.01271, 2018.

[48] Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In European conference on computer vision, pages 649–666. Springer, 2016.

[49] Philip Bachman, R Devon Hjelm, and William Buchwalter. Learning representations by maximizing mutual information across views. Advances in neural information processing systems, 32, 2019.

[50] Xiang Lan, Dianwen Ng, Shenda Hong, and Mengling Feng. Intra-inter subject self-supervised learning for multivariate cardiac signals. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 36, pages 4532–4540, 2022.

Appendix A A Medical Time Series Example

Here we use EEG data of Alzheimer's as an example. An EEG dataset has many patients with or without Alzheimer's (healthy control). Data collectors take multiple trials on a specific patient. These trials could be collected continuously within a short time or across a long period but follow the same experiment manner. Usually, the timestamps are too long for the deep learning pipeline to learn. For example, a 5 minutes trial with a sampling rate 256Hz has 76800 timestamps. Researchers generally use data preprocessing to split a trial into many samples, such as 1-second and 3-second short samples, for further representation learning. Each observation denotes a scalar or a vector of real value at a specific timestamp. An experiment with a sampling rate 256Hz has 256 observations in one second.

Appendix B Data Augmentation Banks

Binomial masking: Generate a mask following a binomial distribution that masks some timestamps of a sample, setting all channels at the masked timestamps to zero.

Channel binomial masking: Generate a mask following a binomial distribution that masks some channels of some timestamps of a sample, setting only a subset of channels at the masked timestamps to zero.

Continuous masking: Mask some continuous sequences of timestamps of a sample, setting all channels at the masked timestamps to zero.

Channel continuous masking: Mask some continuous sequences of timestamps of a sample, setting only a random half of the channels at the masked timestamps to zero.

All true: Do not apply any masking to the sample. Output the raw sample.

Appendix C Shuffle Function Banks

In real-world scenarios, ensuring the presence of samples from the same trial or patient within a training batch becomes increasingly low probability as the number of patients grows. This situation can hinder learning meaningful representations at the trial and patient levels. To address this situation, we have designed two distinct shuffle functions that serve to rearrange samples while also upholding the requirement to include samples originating from the same trial and patient. These functions are called the "trial shuffle" and the "batch shuffle".

Trial shuffle: This function shuffles samples originating from the same trial and subsequently shuffles the trial order. Initially, we arrange the samples by sorting them based on their trial IDs. Next, samples from the same trial are grouped into sets, and the order of samples within each trial set is shuffled. Finally, we sort the trial sets themselves while preserving the order of samples within each respective trial set.

Batch shuffle: This function shuffles samples in a batch and subsequently shuffles the order of batches. The logic of trial and batch shuffle are similar. Initially, we arrange the samples by sorting them based on their trial IDs. Next, we group samples into batch sets, and the order of samples within each batch set is shuffled. Finally, we sort the batch sets themselves while preserving the order of samples within each respective batch set.

Random shuffle: Besides the two specifically designed shuffle functions, this random shuffle function shuffles all the samples in the dataset.

All the shuffle functions mentioned here are designed to shuffle samples within the dataset before training. During training, it is also essential to shuffle samples each epoch to prevent the model from memorizing the dataset and encourage it to learn more useful representations. To address this, we implemented a specially crafted BatchSampler class in PyTorch, following the "batch shuffle" approach. This BatchSampler shuffles the samples locally within each epoch, ensuring that the pre-shuffled sample order is not disrupted significantly. This approach guarantees that each batch contains samples from the same trial. It's worth noting that when a batch consists of samples from the same trial, it must have samples from the same patient.

Appendix D Data Preprocessing

D.1 AD Data Preprocessing

The AD dataset [44] comprises EEG recordings from 12 patients with Alzheimer's disease and 11 healthy controls. Each patient has an average of 30.0 ± 12.5 trials. Each trial corresponds to a 5-second interval with 1280 timestamps (sampled at 256Hz) and includes 16 channels. Prior to further processing, each trial is scaled using a standard scaler. To facilitate analysis, we segment each trial into nine half-overlapping samples, where each sample has a duration of 256 timestamps (equivalent to 1 second). Additionally, we assign a unique trial ID and patient ID to each sample based on its origin in terms of the patient and trial. We split training, validation, and test sets in a patient-independent way. We use samples from patient IDs 17 and 18 as the validation set and samples from IDs 19 and 20 as the test set. The rest of the samples are all put into the training set.

D.2 PTB Data Preprocessing

The PTB dataset [45] consists of ECG recordings from 290 patients, with 15 channels sampled at 1000 Hz. There are a total of 8 types of heart diseases present in the dataset. For this paper, we focus on binary classification using a subset of the dataset that includes 198 patients with major disease labels, specifically Myocardial infarction and healthy controls. To preprocess the ECG signals, they are first normalized using a standard scaler after being resampled to a frequency of 250 Hz. Due to special peak information in ECG signals, a regular sliding window segmentation approach may result in the loss of crucial information. To address this issue, a different segmentation strategy is employed. Instead of sliding windows, the raw trials are segmented into individual heartbeats, with each heartbeat treated as a sample. To perform this segmentation, (1) the first step involves determining the maximum duration. The median value of R-Peak intervals across all channels is calculated for each raw trial, and outliers are removed to obtain a reasonable maximum interval as the standard heartbeat duration. (2) The next step is to determine the position of the first R-Peak. The median value of the first R-Peak position is calculated and used for all channels. (3) Next, the raw trials are split into individual heartbeat segments based on the median value of their respective R-Peak intervals. Each heartbeat is sampled starting from the R-Peak position, with the segments extending to both ends with half the length of the median interval. (4) To ensure the same length of the heartbeat samples, zero padding is applied to align their lengths with the maximum duration. (5) Finally, the samples are merged into trials, where 10 nearby samples are grouped together to form a pseudo-trial, similar to the neighborhood idea presented in [17]. We split training, validation, and test sets in a patient-independent way. We use samples from 28 patients(7 healthy and 21 positive) as the validation set and samples from another 28 patients(7 healthy and 21 positive) as the test set. The rest of the samples are all put into the training set.

D.3 TDBrain Data Preprocessing

The TDBrain [46] is a large dataset that monitors the brain signals of 1274 patients with 33 channels (500 Hz) during EC (Eye closed) and EO (Eye open) tasks. The dataset consists of 60 types of diseases, and it is possible for a patient to have multiple diseases simultaneously. This paper focuses on a subset of the dataset, specifically 25 patients with Parkinson's disease and 25 healthy controls. Only the EC task trials are used for representation learning. To process the raw EC trials, we employ a sliding window approach that continuously moves from the middle of the trial to both ends without any overlap. Each raw EC trial is divided into processed pseudo-trials with a length of 2560 timestamps (10 seconds) after resampling to 256 Hz. These processed pseudo-trials are then scaled using a standard scaler. Furthermore, each pseudo-trial is split into 19 half-overlapping samples, with each sample having a length of 256 timestamps (1 second). In addition to the binary label indicating Parkinson's disease or healthy, each sample is assigned a patient and trial ID based on the patient and processed trial it originates from. It is important to note that the trial ID refers to the ID of the processed pseudo-trial and not the raw EC trial. We split training, validation, and test sets in a patient-independent way. We use samples from 8 patients(4 healthy and 4 positive) as the validation set and samples from another 8 patients(4 healthy and 4 positive) as the test set. The rest of the samples are all put into the training set.

Appendix E COMET and Baseline Implementation Details

We implement the baselines following their corresponding papers, including TS2vec [13], Mixing-up [16], TS-TCC [27], SimCLR [43], CLOCS [15], and TF-C [26]. In our COMET framework and all baselines, we use the last epoch of the contrastive pre-training encoder G for downstream tasks. For the P-FT and F-FT tasks, we save the best model in terms of F1 score on the validation set during training and load the saved model to evaluate the performance on the test set.

COMET (our model) We use a two-layer, fully connected network as the projection head to map the input dimension to the output dimension. The input dimension corresponds to the feature dimension of the data, while the hidden dimension is set to 128, and the output dimension is set to 64. To augment the data, we apply time series masking on the output dimension using the [all_true, all_true, continuous, continuous] (see Appendix B) for the observation, sample, trial, and patient-level contrastive blocks, respectively. The augmented output dimension from the projection head is then passed to the encoder G for representation learning. For the encoder G, we adopt a dilated CNN module. It consists of 10 hidden blocks, each following the order "GELU -> DilatedConv -> GELU -> DilatedConv." A residual connection is applied between the beginning and end of each block. The dilation factor of the convolution in the i-th block is set to 2^i. Each hidden dimension of the dilated convolution is set to 64, and the kernel size is set to 3. The output dimension of encoder G is fixed at 320. We utilize positive pair selection strategies specific to each contrastive block to build the contrastive loss in the embedding space (after encoder G). A max-pooling layer is employed before applying the representation to downstream tasks. During contrastive pre-training, we set the learning rate to 0.0001. The pre-training batch size is 256, and the total number of pre-training epochs is 100. We report the hyperparameters that achieved good results and stability among random seeds. The hyperparameters λ_1, λ_2, λ_3, λ_4 are assigned values of (0.25, 0.25, 0.25, 0.25), (0.1, 0.7, 0.1, 0.1), and (0.25, 0.25, 0.25, 0.25) for the AD, PTB, and TDBrain datasets, respectively.

TS2vec [13] introduces contextual consistency using overlapping subseries and a hierarchical loss function to capture data consistency at the observation and sample levels. To incorporate their methodology, we utilize the open-source code available at https://github.com/yuezhihan/ts2vec. However, their code does not include downstream tasks for F-FT (Full Fine-Tuning), so we implement these tasks using the same setup as our COMET. Specifically, we set the number of epochs for contrastive pre-training to 100, the learning rate to 0.0001, and the batch size to 256. To align with our model, we adjust the convolution blocks to 10, matching our configuration. We adopt the default settings provided by the TS2vec implementation for other settings during pre-training.

TS-TCC [27] leverages temporal and contextual consistency by contrasting a strong and weak augmentation. They employ a transformer-based autoregressive as the encoder. They perform a cross-view prediction by using temporal context to predict one view's future. Then, they maximize the similarity of contexts generated by the encoder to leverage contextual consistency. We utilize the open-source code available https://github.com/emadeldeen24/TS-TCC. We set the number of pre-training epochs to 100. We adopt the default settings provided by the TS-TCC implementation for other settings during pre-training.

Mixing-up [16] proposes a mixing-up augmentation by mixing the proportion of two time-series samples. This augmentation involves creating an augmented sample as the convex combination of two randomly selected time-series samples from the dataset. The mixing parameter follows a beta distribution, determining the proportion of the two samples in the augmentation process. We utilize the open-source code available at https://github.com/wickstrom/mixupcontrastivelearning to implement Mixing-up. Although they only provide downstream tasks for F-FT, we align their method with our setups for all downstream tasks, including P-FT and F-FT. We set the number of pre-training epochs to 100, the learning rate to 0.0001, and the batch size to 256 during pre-training.

SimCLR [18] is the most classic contrastive learning framework first proposed in the CV domain. It applies data augmentation techniques to create augmented views of input samples and constructs a contrastive loss based on these views. While initially designed for images, SimCLR has also been successfully adapted to time series data, as demonstrated in previous work such as [43]. To implement SimCLR, we utilize their open-source code available at https://github.com/iantangc/ContrastiveLearningHAR. We set the number of pre-training epochs to 100, the learning rate to 0.0001, and the batch size to 512, aligning with their recommended settings. We use the default values provided in the SimCLR implementation for other configuration parameters during pre-training.

CLOCS [15] employs samples from the same patient as positive pairs to leverage the data invariance in ECG recordings. They make use of both temporal and spatial information for contrastive learning. To implement CLOCS, we utilized their open-source code, available at https://github.com/danikiyasseh/CLOCS. We incorporated their contrastive loss function to implement the Contrastive Multi-segment Coding (CMSC) mechanism described in their paper. Additionally, we modified their backbone to use TCN, which shares the same network structure as our COMET, including the configuration parameters. Specifically, we set the number of pre-training epochs to 100, the learning rate to 0.0001, and the batch size to 256.

TF-C [26] leverages the consistency between time domain and frequency domain. They assume that the time-based and frequency-based representations of the same example exhibit proximity in the time-frequency space. We utilize their open-source code available at https://github.com/mims-harvard/TFC-pretraining to implement TF-C. While the original method is primarily designed for transfer learning, we extend it to incorporate downstream tasks such as P-FT and F-FT in our experiments. We set the number of pre-training epochs to 100, the learning rate to 0.0001, and the batch size to 256. We use the default values provided in the TF-C implementation for other configuration parameters during pre-training.

E.1 Partial Fine-tuning

In the P-FT (Partial Fine-Tuning) setup, we introduce a classifier L on top of the pre-trained encoder G, while keeping the parameters of G fixed. Only the classifier L is fine-tuned in this setup.

In the COMET, TS2vec, and Mixing-up approaches, we utilize logistic regression from the Sklearn library to implement the classifier L. We use the default settings of Sklearn, except for setting the maximum iteration to 100,000. We employ a one-layer fully connected network as the classifier L for the TF-C method. The learning rates are specifically set to 8e-5, 3e-5, and 1e-4 for the AD, PTB, and TDBrain datasets, respectively. As for TS-TCC and SimCLR, we follow their respective default settings for the partial fine-tuning phase.

E.2 Full Fine-tuning

In the F-FT (Full Fine-Tuning) setup, we introduce a classifier P on top of the pre-trained encoder G, where both the parameters of the encoder G and the classifier P are trainable. In this setup, we fine-tune both the classifier P and the encoder G. We utilize a fraction of 100%, 10%, and 1% labeled training data for fine-tuning.

In the COMET, TS2vec, and Mixing-up approaches, we set the finetune learning rate to 1e-4. We use a batch size of 128 and perform fine-tuning for 50, 100, and 100 epochs for 100%, 10%, and 1% label fractions, respectively. The classifier P is implemented as a two-layer fully connected network with hidden dimensions 128. For TF-C, we change the hidden dimension of this two-layer fully connected network to 64. The learning rates are specifically set to 8e-5, 3e-5, and 1e-4 for the AD, PTB, and TDBrain datasets, respectively. Regarding TS-TCC and SimCLR, we follow their respective default settings for the full fine-tuning phase.

Appendix F Experimental Setting

F.1 Patient-Independent Experimental Setting

This paper adopts a patient-independent setting for the train-validation-test split. In medical time series classification tasks, two common approaches to splitting the data are patient-independent and patient-dependent settings [15, S1]. Figure 3 illustrates these two settings' differences. In the patient-dependent setting, samples from the same patient can appear in both the training and testing sets, whereas in the patient-independent setting, samples from the same patient are exclusively included in either the training or testing set.

Performing patient-independent representation learning poses challenges due to the unique characteristics and different data distributions exhibited by each patient [50]. Even if two patients share the same label, the patterns within their data may differ significantly due to individual noise characteristics, potentially overshadowing the common patterns observed across patients. However, in real-world scenarios, it is essential for a model to be robust and general to perform patient-independent

representation learning. The goal is to train a model on a subset of patients with known labels and utilize it to predict other patients with unknown labels. In contrast, patient-dependent classification is usually impractical since it requires knowledge of the labels for all patients.

F.2 Pseudo-Trial Experimental Setting

For many medical time series datasets, patient information is readily available, but trial information may be absent or limited to a single trial per patient. In such cases, the question arises of effectively employing trial-level contrastive learning. The solution is straightforward. We can generate pseudo-trials rather than merely deactivating the trial-level block by setting λ to 0.

We can define groups of adjacent samples as pseudo-trials and assign them the same trial ID. In our paper, we employed ten neighboring samples as a pseudo-trial for the PTB and TDBrain datasets, and this approach yielded favorable results. This concept is akin to the approach taken by TNC [17], where close samples are defined as positive pairs.

Appendix G Ablation Study, Visualization, Additional Downstream Tasks, and Heavy Duty Baseline

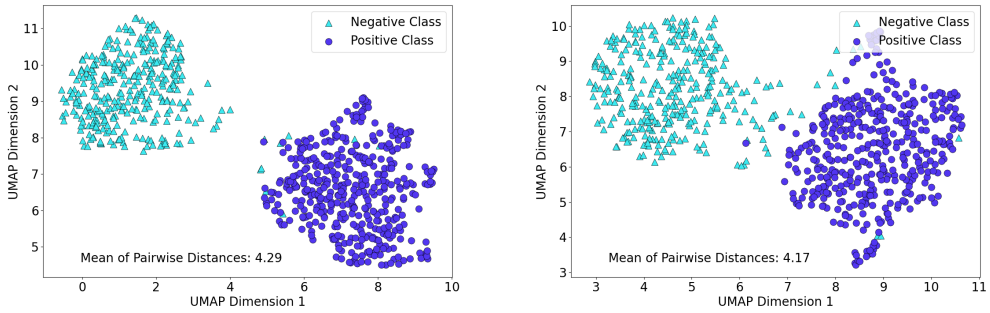
G.1 Ablation Study

Ablation study of contrastive blocks We examined the effectiveness of each contrastive block and progressively incorporated each block from the observation-level to all four levels. Table 5 compares the full-level COMET model with its six variants on the AD, PTB, and TDBrain datasets. The variants are as follows: (1) **O**, **S**, **R**, **P**, which utilize only one level of the contrastive block. We activate that specific level by setting its λ to 1 and deactivate the other three levels by setting their λ to 0; (2) **S+O**, which combines the sample and observation-level contrastive blocks. The λ values for sample and observation levels are set equally to 0.5; (3) **R+S+O**, incorporating the trial, sample, and observation-level contrastive blocks. The λ values for these three levels are set equally to 0.33; and (4) **P+R+S+O**, representing our complete COMET method with all four levels of contrastive blocks. The λ values for the four levels are set equally to 0.25. The "Fraction" column indicates the fraction of labeled training samples used during fine-tuning.

We observed that variants **R** and **P** exhibit strong performance, achieving comparable or even outperforming the full-level COMET model **P+R+S+O** on the PTB and TDBrain datasets across different fractions of labeled data (100%, 10%, and 1%). Notably, they also perform well with a 1% fraction in the AD dataset, although they exhibit significant instability in this case. This finding suggests that adding more contrastive blocks may not necessarily improve final results. Achieving a balance in weights among different levels through λ is crucial. Nevertheless, training models at individual levels can provide insights into the importance of each level in contrastive representation learning. Furthermore, it's noteworthy that the full-level COMET model **P+R+S+O** consistently demonstrates comparable or superior results across all datasets and fractions. Importantly, we did not perform any parameter tuning here, opting to set all the λ values equally, highlighting the stability of the full-level COMET **P+R+S+O** compared to other variants.

Ablation study of hyperparameter λ We conducted an analysis to assess the impact of hyper-parameter λ on the AD, PTB, and TDBrain datasets in table 6. The values of $\lambda_1, \lambda_2, \lambda_3$, and λ_4, from left to right, correspond to the patient, trial, sample, and observation levels, respectively. In this analysis, all four data levels are incorporated, representing the full-level COMET model **P+R+S+O**. We applied a significant weight to one data level by setting its corresponding λ to 0.7 while assigning lower weights of 0.1 to the other levels. Furthermore, we explored scenarios with increased weights on patient and trial levels or sample and observation levels. The "Fraction" column indicates the fraction of labeled training samples used during fine-tuning.

We observed that the results exhibited greater stability than the ablation study of contrastive blocks. In the contrastive block ablation study, there were significant discrepancies between different COMET variants at times. For instance, the **S** and **R** variants of the TDBrain dataset exhibited substantial differences across various fraction setups. However, in the full-level COMET model, the differences between these inter-running setups were notably reduced, even when a heavy weight was applied to one data level. For instance, the differences between lambda setups (0.1, 0.1, 0.7, 0.1) and (0.1,



(a) COMET(Ours) **(b)** TS2vec

Figure 4: Visualizing the learned representation (a) Visualization for TS2vec. (b) Visualization for COMET(Ours). The visualized representation is trained in the F-FT setup on the AD dataset. Dark blue and light blue denotes the negative class(Health) and positive class(Alzheimer), respectively. We calculate the mean Euclidean distance between pairwise samples from two classes for each pair of samples to evaluate the class separability. As the figure shows, our method exhibits superior separation between the two classes, resulting in larger pairwise distances.

0.7, 0.1, 0.1) were not as significant in the TDBrain dataset. This observation underscores again the stability and robustness of the full-level COMET model.

G.2 Visualization

To visualize the effectiveness of COMET, we depict the learned representation h_i using the F-FT setup on the AD dataset as a case study. It is important to note that the learned representation h_i consists of 320 dimensions after pooling. To visualize the representations more interpretably, we employ UMAP, a dimensionality reduction technique with 20 neighbors and a minimum distance of 0.2. To establish a benchmark for comparison, we utilize TS2vec, which has shown the best performance among all baselines in the F-FT setup on the AD dataset. Additionally, we compute the average pairwise Euclidean distance between the negative (healthy) and positive (Alzheimer) classes, offering a quantitative measure of separability between the two classes.

G.3 Performance on Downstream Tasks

Clustering We assess the clustering performance of COMET using the AD dataset as a case study. Instead of employing a classifier model on top of the encoder, we apply K-means clustering (K=2) to the encoder G. We utilize three widely-used evaluation metrics: Silhouette score, Adjusted Rand Index (ARI), and Normalized Mutual Information (NMI). To establish a benchmark for comparison, we consider TS2vec, which has shown the best performance among all baselines in the F-FT setup on the AD dataset. Table 7 illustrates that COMET surpasses TS2vec with an improvement of 0.0586 in Silhouette score, 0.945 in ARI, and 0.881 in NMI.

Anomaly detection We evaluate the anomaly detection performance of COMET using the AD dataset as a case study. While some previous works focus on identifying outlier observations within a sample [S2, S3], we concentrate on sample-level anomaly detection. We construct a very unbalanced AD test set comprising 90% negative (healthy) samples and 10% positive (Alzheimer's) samples. The negative samples are considered normal, while the positive samples are treated as outliers. The test set is prepared accordingly, while the remaining aspects of the experiment follow the F-FT setup. Specifically, We utilize the saved trained models from the F-FT setup to evaluate the new test set, and for comparison, we still employ TS2vec as a benchmark. The "Fraction" column indicates the fraction of labeled training samples used during fine-tuning. The experiment result is shown in table 8. The COMET outperforms TS2vec by 5.25%, 15.3% and 11.4% with label fraction 100%, 10% and 1%, respectively.

Table 5: Ablation study of contrastive blocks. The ablation study of contrastive blocks is evaluated on the AD, TDBrain, and PTB datasets. We examine the effectiveness of each contrastive block. Besides, we progressively incorporate each block from the observation-level to all four levels. Here, **O**, **S**, **R**, and **P** denote the observation, sample, trial, and patient-level contrastive blocks, respectively.

Datasets	Fraction	Blocks	Accuracy	Precision	Recall	F1 score	AUROC	AUPRC
AD	100%	O	81.69±10.71	87.28±5.71	79.64±12.07	78.90±13.78	92.54±4.18	92.36±4.34
		S	83.53±2.89	84.35±1.69	82.81±3.56	82.97±3.47	91.01±1.96	90.81±1.95
		R	78.58±14.99	83.14±11.60	76.31±16.66	74.53±18.69	85.05±12.74	84.44±13.22
		P	72.69±13.13	78.67±11.81	69.74±14.50	67.04±17.27	83.30±13.91	82.57±14.29
		S+O	**85.70±1.82**	86.14±1.63	**85.09±2.04**	**85.35±1.96**	92.21±1.40	92.09±1.39
		R+S+O	85.57±4.04	88.12±2.58	84.31±4.79	84.73±4.59	93.28±2.52	92.98±2.83
		P+R+S+O	84.50±4.46	**88.31±2.42**	82.95±5.39	83.33±5.15	**94.44±2.37**	**94.43±2.48**
	10%	O	86.35±9.25	87.09±9.04	85.55±9.92	85.70±10.10	92.62±8.99	92.77±8.72
		S	77.30±3.63	78.45±3.75	76.26±3.91	76.36±3.92	85.43±3.61	84.63±3.76
		R	77.83±17.59	83.45±14.35	75.41±19.62	72.08±23.45	83.51±16.90	83.27±17.00
		P	71.41±15.31	76.91±15.56	68.70±16.46	66.54±17.74	76.11±16.98	76.06±16.19
		S+O	82.73±2.05	84.33±2.71	81.51±2.07	81.98±2.11	90.02±2.47	90.02±2.40
		R+S+O	91.19±3.14	91.74±3.01	90.70±3.34	90.98±3.25	95.86±2.63	95.86±2.67
		P+R+S+O	**91.43±3.12**	**92.52±2.36**	**90.71±3.56**	**91.14±3.31**	**96.44±2.84**	**96.48±2.82**
	1%	O	69.56±7.54	71.43±8.04	68.19±7.16	67.83±7.41	77.32±8.69	77.32±8.45
		S	59.09±3.50	61.08±4.86	59.83±4.00	58.12±3.57	63.99±6.62	63.33±6.09
		R	**90.15±13.23**	**93.83±7.28**	**89.03±14.88**	**88.17±16.78**	**96.22±5.97**	96.02±6.28
		P	85.57±13.45	90.36±7.16	83.98±15.13	82.72±18.03	94.23±5.82	94.10±5.65
		S+O	63.24±3.62	63.52±4.15	63.30±4.26	62.72±4.14	67.98±5.17	67.25±4.97
		R+S+O	82.65±4.23	83.21±4.06	82.97±4.58	82.46±4.43	90.07±6.77	90.24±6.56
		P+R+S+O	88.22±3.36	88.55±2.73	88.56±3.14	88.14±3.37	96.05±1.36	**96.12±1.31**
PTB	100%	O	85.27±1.73	84.21±1.25	77.74±4.01	79.78±3.37	88.13±2.23	84.76±2.14
		S	84.30±2.56	84.00±2.16	75.68±5.69	77.74±5.13	88.66±2.05	84.80±2.15
		R	88.63±1.43	88.42±1.45	82.46±2.35	84.72±2.12	89.29±3.96	86.21±4.96
		P	**88.85±3.22**	**88.74±2.30**	**82.78±6.11**	**84.75±5.38**	**94.32±1.81**	**90.61±2.81**
		S+O	84.38±2.34	84.33±1.18	75.49±5.69	77.69±4.76	90.37±2.59	86.57±4.32
		R+S+O	85.32±1.93	84.82±1.78	77.54±4.58	79.64±3.94	92.34±1.85	88.75±1.98
		P+R+S+O	86.36±1.44	87.18±1.28	77.87±2.55	80.79±2.44	93.41±2.35	89.09±1.64
	10%	O	86.75±1.44	85.42±2.10	80.88±3.27	82.45±2.29	90.71±3.16	88.84±3.59
		S	86.34±0.73	84.75±1.49	80.21±1.78	81.90±1.25	88.36±0.31	85.20±1.17
		R	88.12±1.75	86.36±2.28	84.16±3.55	84.80±2.39	91.80±2.50	90.45±1.90
		P	**90.38±2.23**	**90.33±2.71**	85.69±4.36	87.27±3.45	**93.06±3.86**	**91.83±2.93**
		S+O	85.84±1.74	84.14±0.76	80.07±5.18	81.14±3.82	90.28±2.75	87.57±3.42
		R+S+O	85.88±3.08	83.60±3.61	82.45±1.94	82.48±2.71	90.79±1.79	88.68±1.91
		P+R+S+O	90.32±1.61	89.67±1.94	**85.85±2.99**	**87.35±2.36**	92.78±1.11	90.46±3.09
	1%	O	85.73±1.59	83.79±2.15	79.53±3.14	81.09±2.54	90.15±2.32	88.85±2.64
		S	77.63±1.72	72.36±2.41	69.41±2.53	70.38±2.29	79.09±2.21	75.07±1.88
		R	86.28±1.97	84.02±2.85	82.06±2.08	82.61±1.97	90.50±0.99	88.32±1.40
		P	**91.55±2.45**	**90.28±3.27**	**88.45±3.22**	**89.25±3.13**	**95.12±2.58**	**94.80±2.04**
		S+O	81.78±1.07	77.65±1.34	77.17±2.14	77.20±1.43	86.65±2.38	82.12±1.95
		R+S+O	80.71±2.28	76.79±2.03	80.51±1.67	77.84±2.13	88.76±1.38	85.00±2.23
		P+R+S+O	86.62±3.90	82.99±4.52	85.36±5.73	83.89±4.96	93.78±3.82	91.94±4.96
TDBrain	100%	O	89.92±2.15	90.50±2.07	89.92±2.15	89.89±2.17	96.79±1.29	96.84±1.29
		S	77.86±2.85	78.98±2.51	77.86±2.85	77.62±3.01	88.44±2.23	88.96±2.14
		R	**94.44±1.79**	**94.60±1.75**	**94.44±1.79**	**94.44±1.80**	**98.39±1.29**	**98.31±1.46**
		P	93.18±3.70	93.27±3.67	93.18±3.70	93.18±3.70	97.59±2.01	97.58±2.00
		S+O	80.38±4.22	81.59±3.70	80.38±4.22	80.16±4.38	91.35±2.24	91.64±2.05
		R+S+O	86.01±4.29	86.35±3.81	86.01±4.29	85.95±4.38	94.14±2.72	94.37±2.58
		P+R+S+O	85.47±1.16	85.68±1.20	85.47±1.16	85.45±1.16	93.73±1.02	93.96±0.99
	10%	O	85.34±4.38	86.03±3.97	85.34±4.38	85.25±4.48	93.18±3.52	93.24±3.53
		S	74.02±2.09	74.62±2.01	74.02±2.09	73.86±2.17	81.92±3.25	81.76±3.42
		R	**92.96±8.22**	**93.02±8.20**	**92.96±8.22**	**92.96±8.23**	**96.14±5.80**	**95.93±6.08**
		P	89.38±14.69	89.58±14.72	89.38±14.69	89.36±14.69	93.23±11.81	93.52±11.10
		S+O	74.92±2.57	76.60±3.07	74.92±2.57	74.54±2.55	84.51±3.07	84.40±3.07
		R+S+O	81.90±4.74	83.55±4.02	81.90±4.74	81.61±4.99	91.21±3.52	90.73±3.72
		P+R+S+O	79.28±4.64	79.83±4.83	79.28±4.64	79.19±4.62	88.39±4.13	88.38±3.96
	1%	O	71.52±7.54	72.17±7.71	71.52±7.54	71.31±7.62	78.32±8.37	77.56±8.91
		S	58.62±3.84	59.66±3.95	58.62±3.84	57.39±4.26	62.85±6.38	61.61±6.61
		R	**85.29±5.93**	**85.55±5.66**	**85.29±5.93**	**85.24±5.99**	**91.19±4.76**	**91.11±4.74**
		P	77.23±3.42	78.10±3.39	77.23±3.42	77.04±3.47	86.12±3.88	85.10±4.11
		S+O	61.71±2.97	61.82±2.90	61.71±2.97	61.60±3.07	66.09±2.98	64.94±2.66
		R+S+O	72.15±6.26	73.39±7.50	72.15±6.26	71.91±6.12	78.39±7.77	76.97±7.61
		P+R+S+O	72.93±7.21	74.20±7.68	72.93±7.21	72.57±7.37	78.72±8.42	77.71±9.10

Table 6: Ablation study of hyperparameter λ. The ablation study of hyperparameter λ is evaluated on the AD, TDBrain, and PTB datasets. The λ_1, λ_2, λ_3, λ_4 from left to right are for patient, trial, sample, and observation levels, respectively.

Datasets	Fraction	$\lambda_1, \lambda_2, \lambda_3, \lambda_4$	Accuracy	Precision	Recall	F1 score	AUROC	AUPRC
AD	100%	(0.1,0.1,0.1,0.7)	**87.82±7.44**	**91.08±4.10**	**86.62±8.58**	**86.77±8.64**	**97.95±1.26**	**97.94±1.26**
		(0.1,0.1,0.7,0.1)	85.03±5.64	89.09±3.22	83.32±6.37	83.78±6.59	95.33±1.81	95.30±1.90
		(0.1,0.7,0.1,0.1)	82.76±8.67	88.00±4.65	80.98±10.02	80.72±10.30	95.32±2.72	95.26±2.71
		(0.7,0.1,0.1,0.1)	82.22±8.31	87.27±4.41	80.42±9.62	80.17±10.00	94.84±2.64	94.80±2.65
		(0.2,0.2,0.3,0.3)	86.88±5.78	89.96±4.00	85.46±6.52	85.97±6.36	94.03±3.56	94.07±3.54
		(0.3,0.3,0.2,0.2)	86.72±5.97	89.76±3.12	85.41±6.92	85.73±6.96	95.14±1.95	95.09±2.04
		(0.3,0.3,0.15,0.25)	82.84±6.96	88.09±3.99	80.87±7.93	81.04±8.06	94.73±2.98	94.64±3.12
	10%	(0.1,0.1,0.1,0.7)	89.56±8.67	91.37±6.73	88.64±9.65	88.82±9.51	96.55±4.41	96.47±4.63
		(0.1,0.1,0.7,0.1)	87.84±6.09	88.72±5.16	87.50±6.76	87.42±6.66	94.58±5.32	94.38±5.67
		(0.1,0.7,0.1,0.1)	79.89±14.06	85.79±7.50	77.97±16.05	75.54±19.81	92.89±6.45	92.67±6.64
		(0.7,0.1,0.1,0.1)	78.26±13.58	84.99±6.68	76.38±15.80	73.44±19.24	92.74±7.64	92.57±7.93
		(0.2,0.2,0.3,0.3)	**93.25±1.71**	**93.86±1.24**	**92.77±2.00**	**93.09±1.80**	**97.68±0.87**	**97.75±0.82**
		(0.3,0.3,0.2,0.2)	91.43±3.70	92.71±2.39	90.68±4.28	91.09±4.00	96.81±1.62	96.88±1.64
		(0.3,0.3,0.15,0.25)	91.54±3.63	92.93±2.32	90.77±4.25	91.19±3.91	97.26±1.63	97.29±1.67
	1%	(0.1,0.1,0.1,0.7)	95.88±2.29	96.21±1.92	95.64±2.55	95.79±2.36	92.72±9.34	93.24±8.74
		(0.1,0.1,0.7,0.1)	85.03±9.80	87.16±5.75	85.65±8.54	84.62±10.42	94.71±2.73	94.73±2.89
		(0.1,0.7,0.1,0.1)	**97.19±1.28**	**97.32±1.11**	**97.08±1.42**	**97.15±1.31**	**99.02±1.06**	**98.98±1.11**
		(0.7,0.1,0.1,0.1)	97.00±1.10	97.10±1.06	96.92±1.14	96.96±1.12	98.93±1.20	98.89±1.26
		(0.2,0.2,0.3,0.3)	84.55±7.39	84.47±7.51	84.75±7.65	84.44±7.49	89.95±10.83	89.78±11.27
		(0.3,0.3,0.2,0.2)	86.96±5.92	87.16±5.66	86.87±6.31	86.72±6.19	92.68±7.76	92.93±7.43
		(0.3,0.3,0.15,0.25)	92.40±2.52	92.75±2.41	92.16±2.44	92.27±2.55	97.76±1.11	97.76±1.17
PTB	100%	(0.1,0.1,0.1,0.7)	85.51±2.46	85.54±2.52	76.96±4.20	79.60±3.91	91.30±2.28	87.11±3.84
		(0.1,0.1,0.7,0.1)	85.94±3.40	87.23±4.03	76.69±5.61	79.64±5.94	93.78±2.51	**90.78±4.78**
		(0.1,0.7,0.1,0.1)	**87.84±1.98**	87.67±1.72	**81.14±3.68**	**83.45±3.22**	92.95±1.56	87.47±2.82
		(0.7,0.1,0.1,0.1)	87.76±2.75	**88.20±1.41**	80.65±5.54	82.96±5.07	91.82±3.66	88.70±3.08
		(0.2,0.2,0.3,0.3)	87.74±1.46	87.78±1.48	80.79±2.58	83.28±2.31	92.97±1.97	88.67±1.71
		(0.3,0.3,0.2,0.2)	87.16±2.02	87.62±1.60	79.47±3.69	82.15±3.33	**94.00±0.42**	88.83±1.65
		(0.3,0.3,0.15,0.25)	87.56±0.87	87.91±0.64	80.25±1.69	82.92±1.50	91.85±2.44	88.27±2.34
	10%	(0.1,0.1,0.1,0.7)	87.76±2.62	87.15±2.27	81.53±5.10	83.43±4.15	92.54±2.17	89.85±2.65
		(0.1,0.1,0.7,0.1)	88.29±2.53	85.80±3.57	**86.87±1.39**	85.89±2.34	94.56±0.80	**93.15±0.75**
		(0.1,0.7,0.1,0.1)	88.49±3.28	88.98±2.60	81.65±6.00	84.01±5.61	**94.83±1.08**	92.48±2.22
		(0.7,0.1,0.1,0.1)	88.70±3.20	**89.47±2.40**	81.85±6.02	84.26±5.45	94.30±1.76	93.24±2.41
		(0.2,0.2,0.3,0.3)	89.25±2.07	88.87±2.55	83.98±3.97	85.72±3.13	94.26±1.47	90.82±3.43
		(0.3,0.3,0.2,0.2)	89.39±2.02	88.86±2.90	84.43±3.65	86.03±3.00	93.59±1.13	92.42±2.00
		(0.3,0.3,0.15,0.25)	**89.82±3.00**	89.36±2.16	84.71±5.73	**86.34±4.84**	93.55±1.96	90.54±3.20
	1%	(0.1,0.1,0.1,0.7)	89.45±1.80	87.49±2.15	86.23±3.61	86.62±2.57	94.03±2.59	93.21±2.60
		(0.1,0.1,0.7,0.1)	85.12±2.24	81.33±2.60	83.06±3.24	82.01±2.76	91.12±2.14	88.96±2.92
		(0.1,0.7,0.1,0.1)	**90.52±1.90**	**88.58±2.93**	88.23±1.98	**88.23±2.10**	95.08±1.50	93.78±1.98
		(0.7,0.1,0.1,0.1)	90.19±1.75	87.88±2.23	88.20±2.70	87.87±2.19	94.82±1.65	93.68±2.24
		(0.2,0.2,0.3,0.3)	85.28±4.39	81.48±5.03	84.10±5.66	82.47±5.28	92.87±3.85	90.34±4.96
		(0.3,0.3,0.2,0.2)	86.96±2.56	83.38±2.89	85.65±3.59	84.32±3.15	94.47±1.90	92.31±2.67
		(0.3,0.3,0.15,0.25)	89.43±1.38	86.42±1.96	**88.86±0.94**	87.37±1.39	**95.32±1.04**	**94.29±1.72**
TDBrain	100%	(0.1,0.1,0.1,0.7)	87.20±3.39	87.40±3.34	87.20±3.39	87.18±3.40	94.84±2.34	94.96±2.33
		(0.1,0.1,0.7,0.1)	**88.53±3.86**	**88.66±3.79**	**88.53±3.86**	**88.52±3.87**	95.57±2.74	95.71±2.66
		(0.1,0.7,0.1,0.1)	87.97±3.26	88.33±3.15	87.97±3.26	87.94±3.28	**95.58±1.92**	**95.77±1.86**
		(0.7,0.1,0.1,0.1)	87.25±2.71	87.54±2.74	87.25±2.71	87.22±2.71	95.14±1.73	95.32±1.70
		(0.2,0.2,0.3,0.3)	85.69±2.42	86.00±2.53	85.69±2.42	85.66±2.41	94.14±1.43	94.31±1.28
		(0.3,0.3,0.2,0.2)	85.76±3.18	86.24±2.75	85.76±3.18	85.69±3.24	94.34±1.79	94.45±1.77
		(0.3,0.3,0.15,0.25)	85.54±4.11	86.30±3.62	85.54±4.11	85.44±4.21	94.22±2.86	94.36±2.80
	10%	(0.1,0.1,0.1,0.7)	76.92±4.84	78.35±3.64	76.92±4.84	76.50±5.33	85.85±3.99	85.36±4.30
		(0.1,0.1,0.7,0.1)	81.91±5.84	82.40±5.82	81.91±5.84	81.83±5.87	90.38±5.11	90.72±5.16
		(0.1,0.7,0.1,0.1)	84.51±4.81	84.90±4.61	84.51±4.81	84.45±4.86	92.58±3.30	92.62±3.09
		(0.7,0.1,0.1,0.1)	**84.92±5.83**	**85.24±5.43**	**84.92±5.83**	**84.86±5.94**	**92.69±4.07**	**92.69±3.81**
		(0.2,0.2,0.3,0.3)	80.79±3.84	82.00±3.76	80.79±3.84	80.59±3.94	89.88±3.50	89.56±3.76
		(0.3,0.3,0.2,0.2)	78.93±3.88	79.78±4.06	78.93±3.88	78.77±3.94	88.64±3.52	88.59±3.24
		(0.3,0.3,0.15,0.25)	79.96±5.63	81.00±5.52	79.96±5.63	79.76±5.76	88.89±4.57	88.70±4.21
	1%	(0.1,0.1,0.1,0.7)	68.68±2.88	70.46±3.56	68.68±2.88	68.02±2.95	73.79±4.80	72.99±5.27
		(0.1,0.1,0.7,0.1)	66.49±8.01	68.46±7.64	66.49±8.01	65.03±9.26	73.30±8.15	72.74±8.32
		(0.1,0.7,0.1,0.1)	73.08±8.61	**74.89±8.18**	73.08±8.61	72.45±8.89	78.39±4.58	77.79±4.18
		(0.7,0.1,0.1,0.1)	73.16±8.72	75.14±8.13	73.16±8.72	72.46±9.05	78.42±4.48	77.76±4.05
		(0.2,0.2,0.3,0.3)	71.55±8.49	72.14±9.14	71.55±8.49	71.44±8.44	77.14±9.24	76.04±9.79
		(0.3,0.3,0.2,0.2)	70.11±8.07	71.68±8.95	70.11±8.07	69.54±8.38	76.24±10.55	75.26±10.84
		(0.3,0.3,0.15,0.25)	**73.59±9.28**	74.53±9.61	**73.59±9.28**	**73.33±9.36**	**79.07±9.67**	**77.97±9.62**

Table 7: Performance on downstream clustering. The clustering performance is evaluated on the AD dataset. We compare the baseline TS2vec, which performs best in the F-FT setup.

Method	Silhouette	ARI	NMI
Random Init.	0.1184± 0.0082	0.1189±0.0664	0.1258±0.0660
TS2vec	0.0795± 0.0032	0.0013± 0.0026	0.0018±0.0016
COMET (Ours)	0.1381± 0.0139	0.9358±0.0264	0.8827±0.0414

Table 8: Performance on anomaly detection. Sample-level anomaly detection on a very unbalanced AD test set comprising 90% negative (healthy) samples and 10% positive (Alzheimer) samples.

Fraction	Models	Accuracy	Precision	Recall	F1 score	AUROC	AUPRC
100%	**TS2vec**	82.11±3.30	66.05±2.45	82.13±3.21	68.70±3.37	91.27±1.47	76.80±3.08
	COMET (Ours)	83.03±11.65	71.76±7.60	90.33±6.31	73.95±11.97	97.99±1.37	91.52±5.52
10%	**TS2vec**	76.05±6.35	62.48±2.90	77.33±4.74	62.67±5.00	86.76±3.95	72.21±5.09
	COMET (Ours)	88.22±2.88	73.22±3.31	92.49±1.73	77.97±3.87	97.91±1.14	92.21±4.43
1%	**TS2vec**	67.24±8.05	57.34±3.12	67.87±6.45	54.35±6.21	72.75±6.82	61.32±5.25
	COMET (Ours)	77.57±4.21	64.72±1.95	84.41±3.38	65.74±3.67	93.13±2.82	79.65±7.65

G.4 Heavy Duty Baselines

In COMET, we incorporate four contrastive blocks to leverage four levels of data consistency, allowing the data to pass through the model four times within one epoch during contrastive pre-training. To ensure that our superior performance is not due to increased data passing, we conduct experiments on the AD dataset with two baselines: SimCLR and TS2vec.

SimCLR utilizes only one contrastive block during training. We employ two strategies to match the data passing number with COMET: (1) Run SimCLR with one contrastive block for four times the original number of epochs in pre-training (400 epochs instead of 100). (2) Duplicate the contrastive blocks, resulting in four SimCLR contrastive blocks. The notation **4E** signifies running the model for four times the original number of epochs, while **4B** indicates the use of four times the number of contrastive blocks compared to the original SimCLR.

TS2vec incorporates two contrastive blocks during training, leading to data passing twice within one epoch. Similarly, we adopt two strategies to align the data passing number with COMET: (1) Run TS2vec for two times the original number of epochs in pre-training (200 epochs instead of 100). (2) Duplicate the contrastive blocks, resulting in four TS2vec contrastive blocks. The notation **2E** denotes running the model for four times the original number of epochs, while **2B** indicates the use of four times the number of contrastive blocks compared to the original TS2vec.

The results are presented in Table 9. We observe that simply increasing the number of epochs or contrastive blocks does not improve performance but rather leads to a decrease in most cases. We speculate that this decrease is caused by overfitting.

Appendix H Broader Impacts

Our approach for self-supervised contrastive learning improves classification performance on target datasets in patient-independent medical diagnosis scenarios. Leveraging different data consistency levels in medical time series is crucial to enable effective and accurate contrastive learning without sufficient labels. Our work will encourage the research community to discover universal frameworks for other practical applications based on time series representation learning. We also hope our work can attract more researchers to the more general problem of hierarchical consistency from other related fields.

From the societal perspective, our work and the line of contrastive learning can promote more efficient use of medical time series with the lack of labels. Specifically, our model has the potential to identify patterns and anomalies that may not be immediately apparent to human experts. This could lead to earlier and more accurate diagnoses, improving patient outcomes and reducing healthcare costs. However, practitioners need to be aware of the limitations of the model.

Table 9: Heavy Duty Baselines. Run more epochs or add more contrastive blocks to SimCLR and TS2vec on the AD dataset.

Fraction	Models	Accuracy	Precision	Recall	F1 score	AUROC	AUPRC
100%	**4E-SimCLR**	56.87±2.51	57.67±4.02	53.31±2.30	48.28±4.63	57.67±4.02	51.97±1.44
	4B-SimCLR	53.92±3.81	51.88±3.25	51.26±3.05	46.92±5.07	51.88±3.25	50.75±1.69
	SimCLR	54.77±1.97	50.15±7.02	50.58±1.92	43.18±4.27	50.15±7.02	50.42±1.06
	2E-TS2vec	76.49±6.10	78.97±3.54	77.69±5.16	76.21±6.45	88.44±2.40	88.12±2.53
	2B-TS2vec	81.61±1.65	81.47±1.75	81.53±1.68	81.43±1.65	89.50±1.60	89.22±1.75
	TS2vec	81.26±2.08	81.21±2.14	81.34±2.04	81.12±2.06	89.20±1.76	88.94±1.85
	COMET (Ours)	**84.50**±**4.46**	**88.31**±**2.42**	**82.95**±**5.39**	**83.33**±**5.15**	**94.44**±**2.37**	**94.43**±**2.48**
10%	**4E-SimCLR**	57.97±1.74	58.41±8.31	53.69±2.25	46.47±5.71	58.41±8.31	52.33±1.38
	4B-SimCLR	53.57±6.29	53.89±5.05	52.97±4.22	50.60±5.46	53.89±5.05	51.80±2.34
	SimCLR	56.09±2.25	53.81±5.74	51.73±2.59	44.10±4.84	53.81±5.74	51.08±1.53
	2E-TS2vec	66.29±7.86	69.92±5.13	67.79±6.44	65.36±8.55	78.54±4.98	77.95±5.29
	2B-TS2vec	72.61±4.46	73.86±4.36	72.98±3.66	72.30±4.24	81.91±4.83	81.74±4.85
	TS2vec	73.28±4.34	74.14±4.33	73.52±3.77	73.00±4.18	81.66±5.20	81.58±5.11
	COMET (Ours)	**91.43**±**3.12**	**92.52**±**2.36**	**90.71**±**3.56**	**91.14**±**3.31**	**96.44**±**2.84**	**96.48**±**2.82**
1%	**4E-SimCLR**	58.07±1.93	57.72±3.50	54.92±1.99	51.93±3.11	57.72±3.50	52.91±1.28
	4B-SimCLR	54.67±5.43	54.86±4.94	54.48±4.64	53.68±4.89	54.86±4.94	52.67±2.68
	SimCLR	55.42±2.43	52.18±5.55	51.37±2.76	45.02±4.79	52.18±5.55	50.87±1.45
	2E-TS2vec	63.56±4.62	64.97±3.53	64.49±3.90	63.28±4.69	70.26±3.55	68.77±3.59
	2B-TS2vec	64.18±4.53	64.26±4.80	64.26±4.80	63.93±4.61	70.07±5.97	68.62±6.25
	TS2vec	64.93±3.53	65.28±3.52	65.14±3.59	64.64±3.58	70.56±5.38	68.97±5.75
	COMET (Ours)	**88.22**±**3.36**	**88.55**±**2.73**	**88.56**±**3.14**	**88.14**±**3.37**	**96.05**±**1.36**	**96.12**±**1.31**

All datasets in this paper are publicly available and are not associated with privacy or security concerns. Furthermore, we have followed guidelines on responsible use specified by the primary authors of the datasets used in the current work.

Appendix References

[S1] Cosimo Ieracitano, Nadia Mammone, Alessia Bramanti, Amir Hussain, and Francesco C Morabito. A convolutional neural network approach for classification of dementia stages based on 2d-spectral representation of eeg recordings. Neurocomputing, 323:96–107, 2019.

[S2] Ling Yang and Shenda Hong. Unsupervised time-series representation learning with iterative bilinear temporal-spectral fusion. In International Conference on Machine Learning, pages 25038–25054. PMLR, 2022.

[S3] Dennis Bäßler, Tobias Kortus, and Gabriele Gühring. Unsupervised anomaly detection in multivariate time series with online evolving spiking neural networks. Machine Learning, 111(4):1377–1408, 2022.