
Self-Supervised Contrastive Pre-Training for Time Series via Time-Frequency Consistency

Xiang Zhang^{†*}

Harvard University

xiang_zhang@hms.harvard.edu

Ziyuan Zhao^{*}

Harvard University

ziyuanzhao@college.harvard.edu

Theodoros Tsiligkaridis

MIT Lincoln Laboratory

ttsili@ll.mit.edu

Marinka Zitnik

Harvard University

marinka@hms.harvard.edu

Abstract

Pre-training on time series poses a unique challenge due to the potential mismatch between pre-training and target domains, such as shifts in temporal dynamics, fast-evolving trends, and long-range and short-cyclic effects, which can lead to poor downstream performance. While domain adaptation methods can mitigate these shifts, most methods need examples directly from the target domain, making them suboptimal for pre-training. To address this challenge, methods need to accommodate target domains with different temporal dynamics and be capable of doing so without seeing any target examples during pre-training. Relative to other modalities, in time series, we expect that time-based and frequency-based representations of the same example are located close together in the time-frequency space. To this end, we posit that time-frequency consistency (TF-C) — embedding a time-based neighborhood of an example close to its frequency-based neighborhood — is desirable for pre-training. Motivated by TF-C, we define a decomposable pre-training model, where the self-supervised signal is provided by the distance between time and frequency components, each individually trained by contrastive estimation. We evaluate the new method on eight datasets, including electrodiagnostic testing, human activity recognition, mechanical fault detection, and physical status monitoring. Experiments against eight state-of-the-art methods show that TF-C outperforms baselines by 15.4% (F1 score) on average in one-to-one settings (*e.g.*, fine-tuning an EEG-pretrained model on EMG data) and by 8.4% (precision) in challenging one-to-many settings (*e.g.*, fine-tuning an EEG-pretrained model for either hand-gesture recognition or mechanical fault prediction), reflecting the breadth of scenarios that arise in real-world applications. The source code and datasets are available at <https://github.com/mims-harvard/TFC-pretraining>.

1 Introduction

Time series plays important roles in many areas, including clinical diagnosis, traffic analysis, and climate science [1, 2, 3, 4, 5, 6]. While representation learning has considerably advanced analysis of time series [7, 8, 9] more broadly [10], learning generalizable representations for temporal data remains a fundamentally challenging problem [8, 11]. There are numerous immediate benefits from generating such representations, of which pre-training capability is particularly desirable and of

*These authors contributed equally to this work.

[†]Present address: University of North Carolina at Charlotte, xiang.zhang@uncc.edu

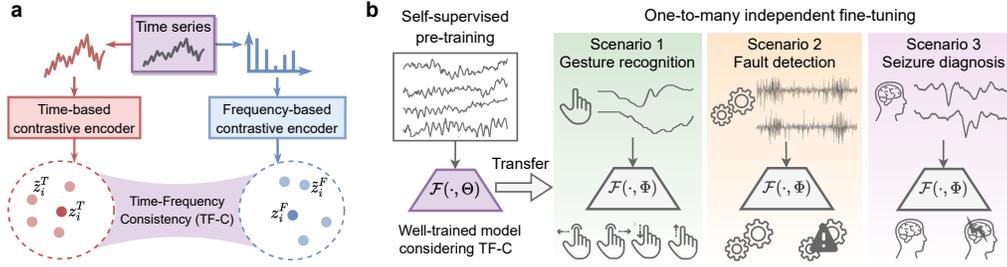


Figure 1: **a.** Illustration of Time-Frequency Consistency (TF-C). Time-based embedding z_i^T and frequency-based embedding z_i^F of time series sample x_i^T , along with \tilde{z}_i^T and \tilde{z}_i^F learned from augmentations of x_i^T , should be close to each other in the latent time-frequency space. **b.** Leveraging TF-C property in time series to optimize a pre-training model \mathcal{F} with parameters Θ that get fine-tuned to Φ on a small scenario-specific dataset.

great practical importance [12, 13]. Central to pre-training is a question of how to process time series in a diverse dataset to greatly improve generalization on new time series coming from different datasets [14, 15, 10]. By training a neural network model on a dataset and transferring it to a new target dataset for fine-tuning, *i.e.*, without explicit retraining on that target data, we expect the resulting performance to be at least as good as that of state-of-the-art models tailored to the target dataset.

However, unfortunately, the expected performance gains are often not realized for a variety of reasons (*e.g.*, distribution shifts, properties of the target dataset unknown during pre-training) [16, 17] that get compounded by the complexity of time series: large variations of temporal dynamics across datasets, varying semantic meaning, irregular sampling, system factors (*e.g.*, different devices or subjects), etc. [18, 17]. This complexity of time series limits the utility of knowledge transfer for pre-training [19, 20]. For example, pre-training a model on a diverse time series dataset with mostly low-frequency components (smooth trends) may not lead to positive transfer on downstream tasks with high-frequency components (transient events) [17]. Examining these challenges can provide clues to what kind of inductive biases could facilitate generalizable representations of time series – this paper offers a strategy for that through a novel time-frequency consistency principle.

In addition, target datasets are not available during pre-training (different from domain adaption [21]; Appendix A), requiring that the pre-training model captures a latent property that holds true for previously unseen target datasets. At the center of this desideratum is the idea of a property that would be shared between pre-training and target datasets and would enable knowledge transfer from pre-training to fine-tuning. In computer vision (CV), pre-training is driven by findings that initial neural layers capture universal visual elements, such as edges and shapes, that are relevant regardless of image style and tasks [22]. In natural language processing (NLP), the foundation for pre-training is given by linguistic principles of semantics and grammar shared across different languages [23]. However, due to the aforementioned temporal complexity, such a principle for pre-training on time series has not yet been established. Moreover, supervised pre-training requires access to large annotated datasets, which limits its use in domains where richly labeled datasets are scarce [24, 25]. For example, in medical applications, labeling data at scale is often infeasible or can be expensive and noisy (experts can disagree on ground-truth labeling [26, 27], *e.g.*, whether an ECG signal indicates a normal vs. abnormal rhythm) [28, 29]. To mitigate these issues, self-supervised learning emerged as a promising strategy to sidestep the lack of labeled datasets [30].

Present work. We introduce a strategy for self-supervised pre-training in time series by modeling Time-Frequency Consistency (TF-C). TF-C specifies that time-based and frequency-based representations, learned from the same time series sample, should be closer to each other in the time-frequency space than representations of different time series samples. Specifically, we adopt contrastive learning in time-space to generate a time-based representation. In parallel, we propose a set of novel augmentations based on the characteristic of the frequency spectrum and produce a frequency-based embedding through contrastive instance discrimination. This is the first work to develop frequency-based contrastive augmentation to leverage rich spectral information and explore time-frequency consistency in time series. The pre-training objective is to minimize the distance between time-based and frequency-based embeddings using a novel consistency loss (Figure 1 (a)). The self-supervised loss is used to optimize the pre-training model and enforce consistency between time and frequency domains in the latent space. The learned relationship encoded in model parameters are transferred to initialize the fine-tuning model and improve performance in datasets of interest (Figure 1 (b)).

We evaluate the TF-C model on eight time series datasets under two evaluation settings (*i.e.*, one-to-one and one-to-many). The eight datasets cover a large set of variations: different numbers of channels (from univariate to 9-channel multivariate), varying time series lengths (from 128 to 5,120), different sampling rates (from 16 Hz to 4,000 Hz), different scenarios (neurological healthcare, human activity recognition, mechanical fault detection, physical status monitoring, etc.) and diverse types of signals (EEG, EMG, ECG, acceleration, and vibration). We compare TF-C approach to eight state-of-the-art baselines. Results show that TF-C achieves positive transfer, outperforming all baselines by a large margin of 15.4% (F1 score) on average. Further, the approach outperforms the strongest baselines with an improvement of up to 7.2% in the F1 score. Finally, the TF-C approach improves prior work by 8.4% in precision (when pre-training the model on sleep EEG signals and fine-tuning it on hand-gesture recognition) in challenging one-to-many setups that apply the same pre-trained model to multiple independent fine-tuning datasets.

2 Related Work

Pre-training for time series. Although there are studies on self-supervised representation learning for time series [7, 8, 31, 32] and self-supervised pre-training for images [33, 34, 35, 24], the intersection of these two areas, *i.e.*, self-supervised pre-training for time series, remains underexplored. In time series, it’s not obvious what reasonable assumptions can bridge pre-training and target datasets. Hence, pre-training models in CV [36, 37, 14] and NLP [10, 15, 38] are not directly applicable due to data modality mismatch, and the existing results leave room for improvement [31, 39, 40]. Shi *et al.* [12] developed the only model to date that is explicitly designed for self-supervised time series pre-training. The model captures the local and global temporal pattern, but it is not convincing why the designed pretext task can capture generalizable representations. Although several studies applied transfer learning in the context of time series [7, 8, 18, 41], there is no foundation yet of which conceptual properties are most suitable for pre-training on time series and why. Addressing this gap, we show that TF-C, designed to be invariant to different time-series datasets, can produce generalizable pre-training models.

Unlike domain adaptation [21, 42] that requires access to target datasets during training, pre-training models do not have access to fine-tuning datasets. As a result, one needs to identify a generalizable time-series property to benefit from pre-training. Further, self-supervised domain adaptation does not need labels in the target dataset but still requires labels for model training [43, 44]. In contrast, TF-C does not need any labels during pre-training.

Contrastive learning with time series. Contrastive learning, a popular type of self-supervised learning, aims to learn an encoder that maps inputs into an embedding space such that positive sample pairs (original augmentation and another alternative augmentation/view of the same input sample) are pulled closer and negative sample pairs (original augmentation and an alternative input sample augmentation) are pushed apart [30, 45]. Contrastive learning in time series is less investigated in comparison, partly due to the challenge of identifying augmentations that capture key invariance properties in time series data. For example, CLOCS defines adjacent time segments as positive pairs [41], and TNC assumes overlapping temporal neighborhoods have similar representations [46]. These methods leverage temporal invariance to define positive pairs which are used to calculate contrastive loss, but other invariances, such as transformation invariance (*e.g.*, SimCLR [40]), contextual invariance (*e.g.*, TS2vec [47] and TS-TCC [48]) and augmentations are possible. In this work, we propose an augmentation bank that exploits multiple invariances to generate diverse augmentations (Sec. 4.1), which adds richness to the pre-training model [48]. Importantly, we propose frequency-based augmentations by perturbing the frequency spectrum of time series (*e.g.*, adding or removing the frequency components and manipulating their amplitude; more details in Sec. 4.2) to learn better representations by exposing the model to a local range of frequency variations. In previous work, CoST processes sequential signals through the frequency domain, but the augmentations are still implemented in time space [49]. Similarly, although BTSF [50] involves frequency domain, its data transformation is solely implemented in the time domain using instance-level dropout. Additional commentary on differences between CoST and BTSF is in Appendix B. To the best of our knowledge, this is the first work that directly perturbs the frequency spectrum to leverage frequency-invariance for contrastive learning. Further, we develop a pre-training model that subjects to TF-C upon two individual contrastive encoders.

3 Problem Formulation

We are given a pre-training dataset $\mathcal{D}^{\text{pret}} = \{\mathbf{x}_i^{\text{pret}} \mid i = 1, \dots, N\}$ of unlabeled time series samples where sample $\mathbf{x}_i^{\text{pret}}$ has K^{pret} channels and L^{pret} timestamps. Let $\mathcal{D}^{\text{tune}} = \{(\mathbf{x}_i^{\text{tune}}, y_i) \mid i = 1, \dots, M\}$ be a fine-tuning (*i.e.*, target; target and fine-tuning are used interchangeably) dataset of labeled time series samples, each having K^{tune} channels and L^{tune} timestamps. Furthermore, every sample $\mathbf{x}_i^{\text{tune}}$ is associated with a label $y_i \in \{1, \dots, C\}$, where C is the number of classes. Without loss of generality, in the following descriptions, we focus on univariate (single-channel) time series, while noting that our approach can accommodate multivariate time series of varying lengths across datasets (shown in experiments in Sec. 5.2). We use superscript symbol \sim to denote contrastive augmentations. We note that $\mathbf{x}_i^{\text{T}} \equiv \mathbf{x}_i$ denotes an input time series sample, and \mathbf{x}_i^{F} denotes discrete frequency spectrum of \mathbf{x}_i .

Problem (Self-Supervised Contrastive Pre-Training For Time Series). *Given are an unlabeled pre-training dataset $\mathcal{D}^{\text{pret}}$ with N samples and a target dataset $\mathcal{D}^{\text{tune}}$ with M samples ($M \ll N$). The goal is to use $\mathcal{D}^{\text{pret}}$ to pre-train a model \mathcal{F} so that by fine-tuning model parameters on $\mathcal{D}^{\text{tune}}$, the fine-tuned model produces generalizable representations $\mathbf{z}_i^{\text{tune}} = \mathcal{F}(\mathbf{x}_i^{\text{tune}})$ for every $\mathbf{x}_i^{\text{tune}}$.*

We follow an established setup, *e.g.*, [41]: for pre-training, only the unlabeled dataset $\mathcal{D}^{\text{pret}}$ is available while, for fine-tuning, a small labeled dataset $\mathcal{D}^{\text{tune}}$ can be used. In short, a model \mathcal{F} is pre-trained on the unlabeled time series dataset $\mathcal{D}^{\text{pret}}$ and its optimized model parameters Θ are fine-tuned to go from $\mathcal{F}(\cdot, \Theta)$ to $\mathcal{F}(\cdot, \Phi)$ using the dataset $\mathcal{D}^{\text{tune}}$. The Φ denotes fine-tuned model parameters. Note that this problem (*i.e.*, $\mathcal{D}^{\text{pret}}$ is independent of the target dataset) is distinct from domain adaptation as fine-tuning dataset $\mathcal{D}^{\text{tune}}$ is not accessed during pre-training. As a result, the pre-trained model can be used with many different fine-tuning datasets without re-training.

Rationale for Time-Frequency Consistency (TF-C). The central idea is to identify a general property that is preserved across time series datasets and use it to induce transfer learning for effective pre-training. The time domain shows how sensor readouts change with time, whereas the frequency domain shows how much of the signal lies within each frequency component over the entire spectrum [51]. Explicitly considering the frequency domain can provide an understanding of time series behavior that cannot be directly captured solely in the time domain [52]. However, existing contrastive methods (*e.g.*, [47, 48]) focus exclusively on modeling the time domain and ignore the frequency domain altogether. One can argue that approach is sufficient in the case of high-capacity methods as time and frequency domains are different views of the same data [53], which can be cross-translated using transformation, such as Fourier and inverse Fourier [54, 52]. The relationship between the two domains, grounded in signal processing theory, provides an invariance that is valid regardless of the time series distribution [55, 56] and thus can serve as an inductive bias for pre-training. Appendix C provides a commentary with analogies for images. Approaching this invariance through the lens of representation learning, we next formulate Time-Frequency Consistency (TF-C). The TF-C property postulates there exists a latent time-frequency space such that for every sample \mathbf{x}_i , time-based representation \mathbf{z}_i^{T} and frequency-based representation \mathbf{z}_i^{F} of the same sample, together with their local augmentations (defined later), are close to each other in the latent space.

Representational Time-Frequency Consistency (TF-C). *Let \mathbf{x}_i be a time series and \mathcal{F} be a model satisfying TF-C. Then, time-based representation \mathbf{z}_i^{T} and frequency-based representation \mathbf{z}_i^{F} as well as representations of \mathbf{x}_i 's local augmentations are proximal in the latent time-frequency space.*

Our strategy is to use dataset $\mathcal{D}^{\text{pret}}$ to induce TF-C in \mathcal{F} 's model parameters Θ , which, in turn, are used to initialize the target model on $\mathcal{D}^{\text{tune}}$ and produce generalizable representations for downstream prediction. The invariant nature of TF-C means that the approach can bridge $\mathcal{D}^{\text{pret}}$ and $\mathcal{D}^{\text{tune}}$ even when large discrepancies exist between them (in terms of temporal dynamics, semantic meaning, etc.), providing a vehicle for a general pre-training on time series.

To realize TF-C, our model \mathcal{F} has four components (Figure 2): a time encoder G_{T} , a frequency encoder G_{F} , and two cross-space projectors R_{T} and R_{F} that map time-based and frequency-based representations, respectively, to the same time-frequency space. Together, the four components provide a way to embed \mathbf{x}_i to the latent time-frequency space such that the time-based embedding $\mathbf{z}_i^{\text{T}} = R_{\text{T}}(G_{\text{T}}(\mathbf{x}_i^{\text{T}}))$ and the frequency-based embedding $\mathbf{z}_i^{\text{F}} = R_{\text{F}}(G_{\text{F}}(\mathbf{x}_i^{\text{F}}))$ are close together.

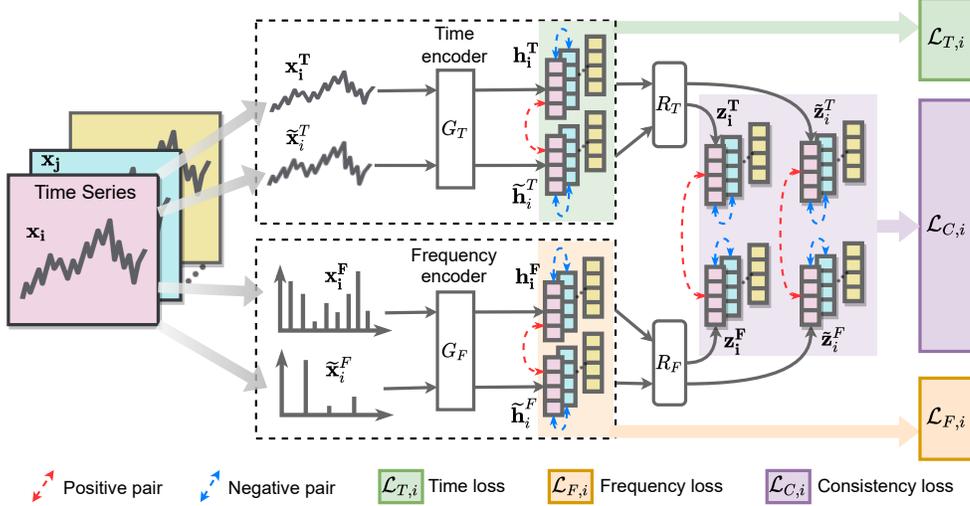


Figure 2: Overview of TF-C approach. Our TF-C pre-training model \mathcal{F} has four components: a time encoder G_T , a frequency encoder G_F , and two cross-space projectors R_T and R_F . For an input time series x_i , the model produces time-based representations (i.e., z_i^T and \tilde{z}_i^T of input x_i and its augmented version, respectively) and frequency-based representations (i.e., z_i^F and \tilde{z}_i^F of input x_i and its augmented version, respectively). The TF-C property is realized by promoting the alignment of time- and frequency-based representations in the latent time-frequency space, providing a vehicle for transferring \mathcal{F} to a target dataset not seen before.

4 Our Approach

Next, we present the architecture of the developed self-supervised contrastive pre-training model \mathcal{F} . Unless specified otherwise, the data mentioned in this section are from pre-training dataset and the superscript pret is omitted for simplification. Here we describe the model using univariate time series as an example, but our model can be straightforwardly applied to multivariate time series (Sec 5).

4.1 Time-based Contrastive Encoder

For a given input time series sample x_i , we generate an augmentation set \mathcal{X}_i^T through a time-based augmentation bank $\mathcal{B}^T : x_i^T \rightarrow \mathcal{X}_i^T$. Each element $\tilde{x}_i^T \in \mathcal{X}_i^T$ is augmented from x_i based on the temporal characteristics. Here, the time-based augmentation bank includes jittering, scaling, time-shifts, and neighborhood segments, all well-established in contrastive learning [40, 48, 41]. We develop an augmentation bank to produce diverse augmentations (rather than a single type of augmentation) and expose the model to complex temporal dynamics, which produces more robust time-based embeddings [48].

For the input x_i , we randomly select an augmented sample $\tilde{x}_i^T \in \mathcal{X}_i^T$ and feed into a contrastive time encoder G_T that maps samples to embeddings. We have $h_i^T = G_T(x_i^T)$ and $\tilde{h}_i^T = G_T(\tilde{x}_i^T)$. As \tilde{x}_i^T is generated based on x_i^T , after passing through G_T , we assume the embedding of x_i^T is close to the embedding of \tilde{x}_i^T but far away from the embedding of x_j^T and \tilde{x}_j^T that are derived from another sample $x_j^T \in \mathcal{D}^{\text{pret}}$ [34, 47, 41]. In specific, we select the positive pair as (x_i^T, \tilde{x}_i^T) and negative pairs as (x_i^T, x_j^T) and (x_i^T, \tilde{x}_j^T) [34].

Contrastive time loss. To maximize the similarity within a positive pair and minimize the similarity within a negative pair, we adopt the NT-Xent (the normalized temperature-scaled cross entropy loss) as distance function d which is widely used in contrastive learning [34, 40]. In specific, we define the loss function of the time-based contrastive encoder in terms of sample x_i^T as:

$$\mathcal{L}_{T,i} = d(h_i^T, \tilde{h}_i^T, \mathcal{D}^{\text{pret}}) = -\log \frac{\exp(\text{sim}(h_i^T, \tilde{h}_i^T)/\tau)}{\sum_{x_j \in \mathcal{D}^{\text{pret}}} \mathbb{1}_{i \neq j} \exp(\text{sim}(h_i^T, G_T(x_j))/\tau)}, \quad (1)$$

where $\text{sim}(u, v) = u^T v / \|u\| \|v\|$ denotes the cosine similarity, the $\mathbb{1}_{i \neq j}$ is an indicator function that equals to 0 when $i = j$ and 1 otherwise, and τ is a temporal parameter to adjust scale. The $x_j \in \mathcal{D}^{\text{pret}}$ refers to a different time series sample or its augmented sample. This loss function

urges the time encoder G_T to generate closer time-based embeddings for positive pairs and push the embeddings for negative pairs apart from each other.

4.2 Frequency-based Contrastive Encoder

We generate the frequency spectrum \mathbf{x}_i^F from a time series sample \mathbf{x}_i^T through a transform operator (e.g., Fourier Transformation [54]). The frequency information in time series is universal and plays a key role in classic signal processing [57, 53, 55], but it is rarely investigated in self-supervised contrastive representation learning for time series [58]. In this section, we develop augmentation method to perturb \mathbf{x}_i^F based on characteristics of frequency spectra and show how to generate frequency-based representations.

As every frequency component in the frequency spectrum denotes a basis function (e.g., sinusoidal function for Fourier transformation) with the corresponding frequency and amplitude, we perturb the frequency spectrum by adding or removing frequency components. A small perturbation in the frequency domain may cause large changes to the temporal patterns in the time domain [55]. To make sure the perturbed time series is still similar to the original sample (not only in frequency domain but also in time domain; Figure 6), we use a small budget E in the perturbations where E denotes the number of frequency components we manipulate. While removing frequency components, we randomly select E frequency components and set their amplitudes to 0. While adding frequency components, we randomly choose E frequency components from the ones have smaller amplitude than $\alpha \cdot A_m$, and increase their amplitude to $\alpha \cdot A_m$. The A_m is the maximum amplitude in the frequency spectrum and α is a pre-defined coefficient to adjust the scale of the perturbed frequency component ($\alpha = 0.5$ in this work). We produce an augmentation set \mathcal{X}_i^F for \mathbf{x}_i^F through frequency-augmentation bank $\mathcal{B}^F: \mathbf{x}_i^F \rightarrow \mathcal{X}_i^F$. As described above, we have two augmentation methods (i.e., removing or adding frequency components) in \mathcal{B}^F , $|\mathcal{X}_i^F| = 2$. Details on the exploration of frequency augmentation strategies are covered in Appendix J.

We utilize a frequency encoder G_F to map the frequency spectrum (e.g., \mathbf{x}_i^F) to a frequency-based embedding (e.g., $\mathbf{h}_i^F = G_F(\mathbf{x}_i^F)$). We assume the frequency encoder G_F can learn similar embedding for the original frequency spectrum \mathbf{x}_i^F and a slightly perturbed frequency spectrum $\tilde{\mathbf{x}}_i^F \in \mathcal{X}_i^F$. Thus, we set the positive pair as $(\mathbf{x}_i^F, \tilde{\mathbf{x}}_i^F)$ and the negative pairs as $(\mathbf{x}_i^F, \mathbf{x}_j^F)$ and $(\mathbf{x}_i^F, \tilde{\mathbf{x}}_j^F)$.

Contrastive frequency loss. We calculate frequency-based contrastive loss for sample \mathbf{x}_i as:

$$\mathcal{L}_{F,i} = d(\mathbf{h}_i^F, \tilde{\mathbf{h}}_i^F, \mathcal{D}^{\text{pret}}) = -\log \frac{\exp(\text{sim}(\mathbf{h}_i^F, \tilde{\mathbf{h}}_i^F)/\tau)}{\sum_{\mathbf{x}_j \in \mathcal{D}^{\text{pret}}} \mathbb{1}_{i \neq j} \exp(\text{sim}(\mathbf{h}_i^F, G_F(\mathbf{x}_j))/\tau)}. \quad (2)$$

In preliminary experiments, we find that the value of τ has little effect on performance and use the same τ throughout all experiments. The $\mathcal{L}_{F,i}$ yield a frequency encoder G_F producing embeddings invariant to frequency spectrum perturbations.

4.3 Time-Frequency Consistency

We develop a consistency loss item $\mathcal{L}_{C,i}$ to urge the learned embeddings to satisfy TF-C: for a given sample, its time-based and frequency-based embeddings (and their local neighborhoods) are supposed to be close to each other (see Sec. 3 for justification). To make sure the distance between embeddings is measurable, we map \mathbf{h}_i^T from time space and \mathbf{h}_i^F from frequency space to a joint time-frequency space through projectors R_T and R_F , respectively. In specific, for every input sample \mathbf{x}_i , we have four embeddings, which are $\mathbf{z}_i^T = R_T(\mathbf{h}_i^T)$, $\tilde{\mathbf{z}}_i^T = R_T(\tilde{\mathbf{h}}_i^T)$, $\mathbf{z}_i^F = R_F(\mathbf{h}_i^F)$, and $\tilde{\mathbf{z}}_i^F = R_F(\tilde{\mathbf{h}}_i^F)$. The first two embeddings are generated based on temporal characteristics and the latter two embeddings are produced based on the properties of frequency spectrum.

To enforce the embeddings in the time-frequency space subject to TF-C, we design a consistency loss $\mathcal{L}_{C,i}$ that measures the distance between a time-based embedding and a frequency-based embedding. We use $S_i^{\text{TF}} = d(\mathbf{z}_i^T, \mathbf{z}_i^F, \mathcal{D}^{\text{pret}})$ to denote the distance between \mathbf{z}_i^T and \mathbf{z}_i^F . Similarly, we define S_i^{TF} , S_i^{TF} , and S_i^{TF} . Note, in this time-frequency space, we don't consider the distance between \mathbf{z}_i^T and $\tilde{\mathbf{z}}_i^T$ where the two embeddings are from the same domain (i.e., time domain). The same applies to pair the distance between \mathbf{z}_i^F and $\tilde{\mathbf{z}}_i^F$. We have already considered information of above two pairs in the calculation of $\mathcal{L}_{T,i}$ and $\mathcal{L}_{F,i}$.

Next, let’s closely observe S_i^{TF} and $S_i^{\widetilde{\text{TF}}}$ that involve three embeddings: z_i^{T} , z_i^{F} , and $\widetilde{z}_i^{\text{F}}$. Here, z_i^{T} and z_i^{F} are learned from the original sample (x_i^{T} and x_i^{F}) while $\widetilde{z}_i^{\text{F}}$ is learned from the augmented $\widetilde{x}_i^{\text{F}}$. Thus, intuitively, z_i^{T} should be closer to z_i^{F} in comparison to $\widetilde{z}_i^{\text{F}}$. Motivated by the relative relationship, we encourage the proposed model to learn a S_i^{TF} that is smaller than $S_i^{\widetilde{\text{TF}}}$. Inspired by the triplet loss [59], we design $(S_i^{\text{TF}} - S_i^{\widetilde{\text{TF}}} + \delta)$ as a term of consistency loss $\mathcal{L}_{\text{C},i}$ where δ is a given constant margin to keep negative samples far apart [60]. This term optimizes the model towards a smaller S_i^{TF} and relatively larger $S_i^{\widetilde{\text{TF}}}$. Similarly, S_i^{TF} is supposed to be smaller than $S_i^{\widetilde{\text{TF}}}$ and $S_i^{\widetilde{\text{TF}}}$. In summary, we calculate the consistency loss $\mathcal{L}_{\text{C},i}$ for sample x_i by:

$$\mathcal{L}_{\text{C},i} = \sum_{S^{\text{pair}}} (S_i^{\text{TF}} - S_i^{\text{pair}} + \delta), \quad S^{\text{pair}} \in \{S_i^{\widetilde{\text{TF}}}, S_i^{\widetilde{\text{TF}}}, S_i^{\widetilde{\text{TF}}}\}, \quad (3)$$

where S_i^{pair} denotes the distance between a time-based embedding (e.g., z_i^{T} or $\widetilde{z}_i^{\text{T}}$) and a frequency-based embedding (e.g., z_i^{F} or $\widetilde{z}_i^{\text{F}}$). In each pair, there is at least one embedding that is derived from augmented sample instead of the original sample. The δ is a pre-defined constant. By combining all the triplet loss items, \mathcal{L}_{C} encourages the pre-training model to capture the consistency between time-based and frequency-based embeddings in model optimization. Note, although the Eq. 3 does not explicitly measure the loss across different time series samples (e.g., x_i and x_j), the cross-sample relationships are implicitly covered in the calculation of S_i^{TF} and S_i^{pair} .

4.4 Implementation and Technical Details

The overall loss function in pre-training has three terms. First, the time-based contrastive loss \mathcal{L}_{T} urges the model to learn embeddings invariant to temporal augmentations. Second, the frequency-based contrastive loss \mathcal{L}_{F} promotes learning of embeddings invariant to frequency spectrum-based augmentations. Third, the consistency loss \mathcal{L}_{C} guides the model to retain the consistency between time-based and frequency-based embeddings. In summary, the pre-training loss is defined as:

$$\mathcal{L}_{\text{TF-C},i} = \lambda(\mathcal{L}_{\text{T},i} + \mathcal{L}_{\text{F},i}) + (1 - \lambda)\mathcal{L}_{\text{C},i} \quad (4)$$

where λ controls the relative importance of the contrastive and consistency losses. We calculate the total loss by summing $\mathcal{L}_{\text{TF-C},i}$ across all pre-training samples. In implementation, the contrastive losses are calculated within the batch. From our problem definition, the model \mathcal{F} we want to learn is the combination of neural networks G_{T} , R_{T} , G_{F} , and R_{F} . When pre-training is completed, we store parameters of entire model, and denote it as $\mathcal{F}(\cdot, \Theta)$ where Θ represents all trainable parameters. When a sample x_i^{tune} is presented, fine-tuned model \mathcal{F} generates an embedding z_i^{tune} via concatenation as: $z_i^{\text{tune}} = \mathcal{F}(x_i^{\text{tune}}, \Phi) = [z_i^{\text{tune},\text{T}}; z_i^{\text{tune},\text{F}}]$ where Φ are fine-tuned model’s parameters.

5 Experiments

We compare the developed TF-C model with 10 baselines on 8 diverse datasets. We investigate the time series classification tasks in the context of one-to-one and one-to-many transfer learning setups (the many-to-one setting is fundamentally different as discussed in Appendix K). We also assess TF-C in extensive downstream tasks including clustering and anomaly detection.

Datasets. (1) **SLEEP EEG** [61] has 371,055 univariate brainwaves (EEG; 100 Hz) collected from 197 individuals. Each sample is associated with one of five sleeping stages. (2) **EPILEPSY** [62] monitors the brain activities of 500 subjects with single-channel EEG sensor (174 Hz). A sample is labeled in binary based on whether the subject has epilepsy or not. (3) **FD-A** [63] gathers the vibration signals from rolling bearing from a mechanical system aiming at fault detection. Every sample has 5,120 timestamps and an indicator for one out of three mechanical device states. (4) **FD-B** [63] has the same setting as the **FD-A** but the rolling bearings are performed in different working conditions (e.g., varying rotational speed). (5) **HAR** [64] has 10,299 9-dimension samples from 6 daily activities. (6) **GESTURE** [65] includes 440 samples that are collected from 8 hand gestures recorded by an accelerometer. (7) **ECG** [26] contains 8,528 single-sensor ECG recordings with sorted into four classes based on human physiology. (8) **EMG** [66] consists of 163 EMG samples with 3-class labels implying muscular diseases. Dataset labels are not used in pre-training. Further dataset statistics are in Appendix D and Table 3.

Baselines. We consider 10 baseline methods. This includes 8 state-of-the-art methods: TS-SD [12], TS2vec [47], CLOCS [41], Mixing-up [18], TS-TCC [48], SimCLR [40], TNC [46], and CPC [30].

Table 1: One-to-one pre-training evaluation (Scenario 3). Pre-training is performed on HAR, followed by fine-tuning on GESTURE. Results for other three scenarios are shown in Tables 4-6.

Models	Accuracy	Precision	Recall	F1 score	AUROC	AUPRC
Non-DL (KNN)	0.6766±0.0000	0.6500±0.0000	0.6821±0.0000	0.6442±0.0000	0.8190±0.0000	0.5231±0.0000
Random Init.	0.4219±0.0865	0.4751±0.0925	0.4963±0.1026	0.4886±0.0967	0.7129±0.1206	0.3358±0.1194
TS-SD	0.6937±0.0533	0.6806±0.0496	0.6883±0.0525	0.6785±0.0495	0.8708±0.0305	0.6261±0.0790
TS2vec	0.6453±0.0260	0.6287±0.0339	0.6451±0.0218	0.6261±0.0294	0.8890±0.0054	0.6670±0.0118
CLOCS	0.4731±0.0229	0.4639±0.0432	0.4766±0.0266	0.4392±0.0198	0.8161±0.0068	0.4916±0.0103
Mixing-up	0.7183±0.0123	0.7001±0.0166	0.7183±0.0123	0.6991±0.0145	0.9127±0.0018	0.7654±0.0071
TS-TCC	0.7593±0.0242	0.7668±0.0257	0.7566±0.0231	0.7457±0.0210	0.8866±0.0040	0.7217±0.0121
SimCLR	0.4383±0.0652	0.4255±0.1072	0.4383±0.0652	0.3713±0.0919	0.7721±0.0559	0.4116±0.0971
TF-C (Ours)	0.7824±0.0237	0.7982±0.0496	0.8011±0.0322	0.7991±0.0296	0.9052±0.0136	0.7861±0.0149

The TS2Vec, TS-TCC, SimCLR, TNC, and CPC are designed for representation learning on a single dataset rather than for transfer learning, so we apply them to fit our settings and make the results comparable. As the training of TNC and CPC are very time-consuming and relatively less competitive (Table 4), we only compare them in the one-to-one setting (scenario 1) while not in other experiments. To examine the utility of pre-training, we consider two additional approaches that are applied directly to fine-tuning datasets without any pre-training: Non-DL (a non-deep learning KNN model) and Random Init. (randomly initializes the fine-tuning model). The evaluation metrics are accuracy, precision (macro-averaged), recall, F1 score, AUROC, and AUPRC.

Implementation. We use two 3-layer 1-D ResNets [67] as backbones for encoders G_T and G_F . Our datasets contain long time series (samples in FD-A and FD-B have 5,120 observations), and preliminary experiments identified ResNet as a better option than a Transformer variant [68]. We use 2 fully-connected layers for R_T and R_F , with no sharing of parameters. We set $E = 1$ and $\alpha = 0.5$ in frequency augmentations and $\tau = 0.2$, $\delta = 1$, $\lambda = 0.5$ in loss functions. Reported are mean and standard deviation values across 5 independent runs (both pre-training and fine-tuning) on the same data split. Results for KNN (K=2) do not change so the standard deviation is zero. Method details and hyper-parameter selection are in Appendix E.

5.1 Results: One-to-One Pre-Training Evaluation

Setup. In one-to-one evaluation, we pre-train a model on *one* pre-training dataset and use it for fine-tuning on *one* target dataset only. **Scenario 1** (SLEEP EEG \rightarrow EPILEPSY): Pre-training is done on SLEEP EEG and fine-tuning on EPILEPSY. While both datasets describe a single-channel EEG, the signals are from different channels/positions on scalps, track different physiology (sleep vs. epilepsy), and are collected from different patients. **Scenario 2** (FD-A \rightarrow FD-B): Datasets describe mechanical devices that operate in different working conditions, including rotational speed, load torque, and radial force. **Scenario 3** (HAR \rightarrow GESTURE): Datasets record different activities (6 types of human daily activities vs. 8 hand gestures). While both datasets contain acceleration signals, HAR has 9 channels while GESTURE has 1 channel. **Scenario 4** (ECG \rightarrow EMG): While both are physiological datasets, the ECG records the electrical signal from the heart whereas EMG measures muscle response in response to a nerve’s stimulation of the muscle. We note that the discrepancies between pre-training and fine-tuning datasets in the above four scenarios are substantial, and they cover a diverse range of variation in time series datasets: varying semantic meaning, sampling frequency, time series length, number of classes, and system factors (*e.g.*, number of devices or subjects). The setup is further challenged by the relatively small number of samples available for fine-tuning (EPILEPSY: 60; FD-B: 60; GESTURE: 480; EMG: 122). Further details are in Appendix F.

Results. The results for the four scenarios are shown in Table 1 and Tables 4-6. Overall, our TF-C model has won 16 out of 24 tests (6 metrics in 4 scenarios) and is the second-best performer in only 8 other tests. We report all metrics but discuss the F1 score in the following. On average, our TF-C model claims a large margin of 15.4% over all baselines. Although the strongest baseline is varying (such as TS-TCC in Scenario 2; Mixing-up in Scenario 3), our model outperforms the strongest baselines by 1.5% across all scenarios. Specifically, as shown in Table 1 (HAR \rightarrow GESTURE; Scenario 3), TF-C achieves the highest performance of 79.91% in F1 score, which yields a margin of 7.2% over the best baseline TS-TCC (74.57%). One potential explanation is that Scenario 3 involves a complex dataset (HAR has 6 classes while GESTURE has 8 classes) that can be difficult to model. The complexity of Scenario 3 is further verified by poor performance of all models ($\pm 80\%$) relative to performance on other Scenarios ($\pm 90\%$): TF-C shows strong robustness by

Table 2: One-to-many pre-training evaluation. Pre-training is performed on SLEEPEEG, followed by an independent fine-tuning on EPILEPSY, FD-B, GESTURE, and EMG.

Scenarios	Models	Accuracy	Precision	Recall	F1 score	AUROC	AUPRC
SLEEPEEG ↓ EPILEPSY	Non-DL (KNN)	0.8525±0.0000	0.8639±0.0000	0.6431±0.0000	0.6791±0.0000	0.6434±0.0000	0.6279±0.0000
	Random Init.	0.8983±0.0656	0.9213±0.1369	0.7447±0.1135	0.7959±0.1208	0.8578±0.2153	0.6489±0.1926
	TS-SD	0.8952±0.0522	0.8018±0.2244	0.7647±0.1485	0.7767±0.1855	0.7677±0.2452	0.7940±0.1825
	TS2vec	0.9395±0.0044	0.9059±0.0116	0.9039±0.0118	0.9045±0.0067	0.9587±0.0086	0.9430±0.0103
	CLOCS	0.9507±0.0027	0.9301±0.0067	0.9127±0.0165	0.9206±0.0066	0.9803±0.0023	0.9609±0.0116
	Mixing-up	0.8021±0.0000	0.4011±0.0000	0.5000±0.0000	0.4451±0.0000	0.9743±0.0081	0.9618±0.0104
	TS-TCC	0.9253±0.0098	0.9451±0.0049	0.8181±0.0257	0.8633±0.0215	0.9842±0.0034	0.9744±0.0043
	SimCLR	0.9071±0.0344	0.9221±0.0166	0.7864±0.1071	0.8178±0.0998	0.9045±0.0539	0.9128±0.0205
	TF-C (Ours)	0.9495±0.0249	0.9456±0.0108	0.8908±0.0216	0.9149±0.0534	0.9811±0.0237	0.9703±0.0199
SLEEPEEG ↓ FD-B	Non-DL (KNN)	0.4473±0.0000	0.2847±0.0000	0.3275±0.0000	0.2284±0.0000	0.4946±0.0000	0.3308±0.0000
	Random Init.	0.4736±0.0623	0.4829±0.0529	0.5235±0.1023	0.4911±0.0590	0.7864±0.0349	0.7528±0.0254
	TS-SD	0.5566±0.0210	0.5710±0.0535	0.6054±0.0272	0.5703±0.0328	0.7196±0.0113	0.5693±0.0532
	TS2vec	0.4790±0.0113	0.4339±0.0092	0.4842±0.0197	0.4389±0.0107	0.6463±0.0130	0.4442±0.0162
	CLOCS	0.4927±0.0310	0.4824±0.0316	0.5873±0.0387	0.4746±0.0485	0.6992±0.0099	0.5501±0.0365
	Mixing-up	0.6789±0.0246	0.7146±0.0343	0.7613±0.0198	0.7273±0.0228	0.8209±0.0035	0.7707±0.0042
	TS-TCC	0.5499±0.0220	0.5279±0.0293	0.6396±0.0178	0.5418±0.0338	0.7329±0.0203	0.5824±0.0468
	SimCLR	0.4917±0.0437	0.5446±0.1024	0.4760±0.0885	0.4224±0.1138	0.6619±0.0219	0.5009±0.0477
	TF-C (Ours)	0.6938±0.0231	0.7559±0.0349	0.7202±0.0257	0.7487±0.0268	0.8965±0.0135	0.7871±0.0267
SLEEPEEG ↓ GESTURE	Non-DL (KNN)	0.6833±0.0000	0.6501±0.0000	0.6833±0.0000	0.6443±0.0000	0.8190±0.0000	0.5232±0.0000
	Random Init.	0.4219±0.0629	0.4751±0.0175	0.4963±0.0679	0.4886±0.0459	0.7129±0.0166	0.3358±0.1439
	TS-SD	0.6922±0.0444	0.6698±0.0472	0.6867±0.0488	0.6656±0.0443	0.8725±0.0324	0.6185±0.0966
	TS2vec	0.6917±0.0333	0.6545±0.0358	0.6854±0.0349	0.6570±0.0392	0.8968±0.0123	0.6989±0.0346
	CLOCS	0.4433±0.0518	0.4237±0.0794	0.4433±0.0518	0.4014±0.0602	0.8073±0.0109	0.4460±0.0384
	Mixing-up	0.6933±0.0231	0.6719±0.0232	0.6933±0.0231	0.6497±0.0306	0.8915±0.0261	0.7279±0.0558
	TS-TCC	0.7188±0.0349	0.7135±0.0352	0.7167±0.0373	0.6984±0.0360	0.9099±0.0085	0.7675±0.0201
	SimCLR	0.4804±0.0594	0.5946±0.1623	0.5411±0.1946	0.4955±0.1870	0.8131±0.0521	0.5076±0.1588
	TF-C (Ours)	0.7642±0.0196	0.7731±0.0355	0.7429±0.0268	0.7572±0.0311	0.9238±0.0159	0.7961±0.0109
SLEEPEEG ↓ EMG	Non-DL (KNN)	0.4390±0.0000	0.3772±0.0000	0.5143±0.0000	0.3979±0.0000	0.6025±0.0000	0.4084±0.0000
	Random Init.	0.7780±0.0729	0.5909±0.0625	0.6667±0.0135	0.6238±0.0267	0.9109±0.1239	0.7771±0.1427
	TS-SD	0.4606±0.0000	0.1545±0.0000	0.3333±0.0000	0.2111±0.0000	0.5005±0.0126	0.3775±0.0110
	TS2vec	0.7854±0.0318	0.8040±0.0750	0.6785±0.0396	0.6766±0.0501	0.9331±0.0164	0.8436±0.0372
	CLOCS	0.6985±0.0323	0.5306±0.0750	0.5354±0.0291	0.5139±0.0409	0.7923±0.0573	0.6484±0.0680
	Mixing-up	0.3024±0.0534	0.1099±0.0126	0.2583±0.0456	0.1541±0.0204	0.4506±0.1718	0.3660±0.1635
	TS-TCC	0.7889±0.0192	0.5851±0.0974	0.6310±0.0991	0.5904±0.0952	0.8851±0.0113	0.7939±0.0386
	SimCLR	0.6146±0.0582	0.5361±0.1724	0.4990±0.1214	0.4708±0.1486	0.7799±0.1344	0.6392±0.1596
	TF-C (Ours)	0.8171±0.0287	0.7265±0.0353	0.8159±0.0289	0.7683±0.0311	0.9152±0.0211	0.8329±0.0137

learning more generalizable representations. Additionally, we visualize the learned representations in time-frequency space (Appendix I), and the analyses provide further support for the TF-C property.

5.2 Results: One-to-Many Pre-Training Evaluation

Setup. In one-to-many evaluation, pre-training is done using *one* dataset followed by fine-tuning on *multiple* target datasets independently without starting pre-training from scratch. Out of eight datasets, SLEEPEEG has most complex temporal dynamics [69] and is the largest (371,055 samples). For that reason, we pre-train a model on SLEEPEEG and separately fine-tune a well-pre-trained model on EPILEPSY, FD-B, GESTURE, and EMG.

Results. Results are shown in Table 2. As there are fewer commonalities between EEG signals vs. vibration, and acceleration vs. EMG, we expect that transfer learning will be less effective for them than one-to-one evaluations. The pre-training and fine-tuning datasets are largely different in the bottom three blocks (SLEEPEEG → {FD-B, GESTURE, EMG}). The large gap reasonably leads to a deterioration in baseline performances, however, our model has a noticeably higher tolerance to knowledge transfer across datasets with large gaps. Notably, We find that the proposed model with TF-C earned the best performance in 14 out of 18 settings in the three challenging settings: indicating our TF-C assumption is universal in time series. For example, our approach outperforms the strongest baseline by 8.4% (in precision) when fine-tuning on GESTURE. Our model has great potential to serve as a universal model when there is no large pre-training dataset that is similar to the small fine-tuning dataset. Furthermore, the TF-C consistently outperforms KNN and Random Init. (which are not pre-trained) by a large margin of 42.8% and 25.1% (both in F1 score) on average.

Ablation study. We evaluate how relevant the model components are for effective pre-training. As shown in Table 9 (SLEEPEEG → GESTURE; Appendix H), removing \mathcal{L}_C , \mathcal{L}_T , and \mathcal{L}_F result in performance degradation (precision) of 6.1%, 7.2%, and 6.7%, respectively. To validate that

the performance increment is not solely brought by a third loss term no matter what consistency it measures, we replaced consistency loss \mathcal{L}_C with a loss term measuring the consistency within time space (named \mathcal{L}_{TT-C}) or within frequency space (named \mathcal{L}_{FF-C}). Results show our consistency loss outperforms \mathcal{L}_{TT-C} and \mathcal{L}_{FF-C} by 5.3% and 7.2% (accuracy), respectively.

5.3 Additional Downstream Tasks: Clustering and Anomaly Detection

Clustering Task. We evaluate the clustering performance of TF-C taking SLEEP EEG \rightarrow EPILEPSY as an example. Specifically, we added a K-means ($K=2$), as Epilepsy has 2 classes, on top of z_i^{tune} in fine-tuning. We adopt commonly used evaluation metrics: Silhouette score, Adjusted Rand Index (ARI), and Normalized Mutual Information (NMI). Table 7 shows our TF-C obtains the best clustering surpassing the strongest baseline (TS-TCC) by a large margin (5.4% in Silhouette score). It conveys that TF-C can capture more distinctive representations with the knowledge transferred from pre-training, which is consistent with the superiority of TF-C in the above classification tasks.

Anomaly Detection Task. We assess how TF-C performs on a sample-level anomaly detection task. Note we work on the sample-level rather than the observation-level anomaly detection. Based on global patterns, the former aims to detect abnormal time series samples instead of outlier observations in a sample (as in BTSF [50] and USAD [70]) which emphasizes local context. Specifically, In the scenario of FD-A \rightarrow FD-B, we built a small subset of FD-B with 1,000 samples, of which 900 are from undamaged bearings, and the remaining 100 are from bearings with inner or outer damage. Undamaged samples are considered “normal,” and inner/outer damaged samples are “outliers.” In fine-tuning, we used one-class SVM on top of learned representations z_i^{tune} . The experimental results (Table 8) show that our TF-C outperforms five competitive baselines with 4.5% in F-1 Score. Results show that the proposed TF-C is more sensitive to anomalous samples and can effectively detect the abnormal status in mechanical devices.

6 Conclusion

We develop a pre-training approach that introduces time-frequency consistency (TF-C) as a mechanism to support knowledge transfer between time-series datasets. The approach uses self-supervised contrastive estimation and injects TF-C into pre-training, bringing time-based and frequency-based representations and their local neighborhoods close together in the latent space.

Limitations and future directions. TF-C property can serve as a universal property for pre-training on diverse time series datasets. Additional generalizable properties, such as temporal autoregressive processes, could also be helpful for pre-training on time series. Further, while our method expects as input a regularly sampled time series, it can handle irregularly sampled time series by using an encoder (such as Raindrop [71] and SeFT [72]) that can embed irregular time series. For frequency encoder inputs x_i^F , alternatives include resampling or interpolation to obtain regularly sampled signals and using regular or non-uniform FFT operations. Furthermore, TF-C’s current embedding strategy and loss functions are favorable for classification, leveraging global information over tasks that use local context (*e.g.*, forecasting). Results show that the TF-C approach performs well across broad downstream tasks, including classification, clustering, and anomaly detection (Sec. 5.3).

Acknowledgments and Disclosure of Funding

We gratefully acknowledge support by US Air Force Contract No. FA8702-15-D-0001, Harvard Data Science Initiative, and awards from Amazon Research, Bayer Early Excellence in Science, AstraZeneca Research, and Roche Alliance with Distinguished Scientists. T.T. is supported by the Under Secretary of Defense for Research and Engineering under US Air Force Contract No. FA8702-15-D-0001. 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] Hrayr Harutyunyan, Hrant Khachatrian, David C Kale, Greg Ver Steeg, and Aram Galstyan. Multitask learning and benchmarking with clinical time series data. *Scientific data*, 6(1):1–18, 2019.
- [2] Shahbaz Rezaei and Xin Liu. Deep learning for encrypted traffic classification: An overview. *IEEE communications magazine*, 57(5):76–81, 2019.
- [3] Suman Ravuri, Karel Lenc, Matthew Willson, Dmitry Kangin, Remi Lam, Piotr Mirowski, Megan Fitzsimons, Maria Athanassiadou, Sheleem Kashem, Sam Madge, et al. Skilful precipitation nowcasting using deep generative models of radar. *Nature*, 597(7878):672–677, 2021.
- [4] Omer Berat Sezer, Mehmet Ugur Gudelek, and Ahmet Murat Ozbayoglu. Financial time series forecasting with deep learning: A systematic literature review: 2005–2019. *Applied soft computing*, 90:106181, 2020.
- [5] Bing Su and Ji-Rong Wen. Temporal alignment prediction for supervised representation learning and few-shot sequence classification. In *ICLR*, 2022.
- [6] Yixiang Deng, Lu Lu, Laura Aponte, Angeliki M Angelidi, Vera Novak, George Em Karniadakis, and Christos S Mantzoros. Deep transfer learning and data augmentation improve glucose levels prediction in type 2 diabetes patients. *NPJ Digital Medicine*, 4(1):1–13, 2021.
- [7] Quentin Rebjock, Baris Kurt, Tim Januschowski, and Laurent Callot. Online false discovery rate control for anomaly detection in time series. *NeurIPS*, 34:26487–26498, 2021.
- [8] Fan-Keng Sun, Chris Lang, and Duane Boning. Adjusting for autocorrelated errors in neural networks for time series. *NeurIPS*, 34:29806–29819, 2021.
- [9] Angus Dempster, François Petitjean, and Geoffrey I Webb. Rocket: exceptionally fast and accurate time series classification using random convolutional kernels. *Data Mining and Knowledge Discovery*, 34(5):1454–1495, 2020.
- [10] Wenyong Huang, Zhenhe Zhang, Yu Ting Yeung, Xin Jiang, and Qun Liu. Spiral: Self-supervised perturbation-invariant representation learning for speech pre-training. *ICLR*, 2022.
- [11] Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. Deep learning for time series classification: a review. *Data mining and knowledge discovery*, 33(4):917–963, 2019.
- [12] Pengxiang Shi, Wenwen Ye, and Zheng Qin. Self-supervised pre-training for time series classification. In *IJCNN*, pages 1–8, 2021.
- [13] Weixia Dang, Biyu Zhou, Lingwei Wei, Weigang Zhang, Ziang Yang, and Songlin Hu. Tsbert: Time series anomaly detection via pre-training model bert. In *International Conference on Computational Science*, pages 209–223. Springer, 2021.
- [14] Soravit Changpinyo, Piyush Sharma, Nan Ding, and Radu Soricut. Conceptual 12m: Pushing web-scale image-text pre-training to recognize long-tail visual concepts. In *CVPR*, pages 3558–3568, 2021.
- [15] Kailai Sun, Zuchao Li, and Hai Zhao. Multilingual pre-training with universal dependency learning. *NeurIPS*, 34:8444–8456, 2021.
- [16] Rui Ye and Qun Dai. Implementing transfer learning across different datasets for time series forecasting. *Pattern Recognition*, 109:107617, 2021.
- [17] Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. Transfer learning for time series classification. In *2018 IEEE international conference on big data (Big Data)*, pages 1367–1376. IEEE, 2018.
- [18] Kristoffer Wickstrøm, Michael Kampffmeyer, Karl Øyvind Mikalsen, and Robert Jenssen. Mixing up contrastive learning: Self-supervised representation learning for time series. *PRL*, 155:54–61, 2022.
- [19] Priyanka Gupta, Pankaj Malhotra, Jyoti Narwariya, Lovekesh Vig, and Gautam Shroff. Transfer learning for clinical time series analysis using deep neural networks. *Journal of Healthcare Informatics Research*, 4(2):112–137, 2020.
- [20] Amiel Meiseles and Lior Rokach. Source model selection for deep learning in the time series domain. *IEEE Access*, 8:6190–6200, 2020.

- [21] Ankit Singh. Clda: Contrastive learning for semi-supervised domain adaptation. *NeurIPS*, 34:5089–5101, 2021.
- [22] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A. Wichmann, and Wieland Brendel. Imagenet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness. In *ICLR*, 2019.
- [23] Alec Radford and Karthik Narasimhan. Improving language understanding by generative pre-training. OpenAI, 2018.
- [24] Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey Hinton. Big self-supervised models are strong semi-supervised learners. In *NeurIPS*, volume 33, pages 22243–22255, 2020.
- [25] Alexei Baevski, Henry Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. In *NeurIPS*, volume 33, pages 12449–12460, 2020.
- [26] Gari D Clifford, Chengyu Liu, Benjamin Moody, H Lehman Li-wei, Ikaro Silva, Qiao Li, AE Johnson, and Roger G Mark. Af classification from a short single lead ecg recording: The physionet/computing in cardiology challenge 2017. In *2017 Computing in Cardiology (CinC)*, pages 1–4. IEEE, 2017.
- [27] Mitchell L Gordon, Kaitlyn Zhou, Kayur Patel, Tatsunori Hashimoto, and Michael S Bernstein. The disagreement deconvolution: Bringing machine learning performance metrics in line with reality. In *CHI*, pages 1–14, 2021.
- [28] Simon Rogers, Derek Sleeman, and John Kinsella. Investigating the disagreement between clinicians’ ratings of patients in icus. *IEEE Journal of Biomedical and Health Informatics*, 17(4):843–852, 2013.
- [29] Leonard M Horowitz, Rita de Sales French, Kirk D Wallis, David L Post, and Ellen Y Siegelman. The prototype as a construct in abnormal psychology: Ii. clarifying disagreement in psychiatric judgments. *Journal of Abnormal Psychology*, 90(6):575, 1981.
- [30] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. In *arXiv:1807.03748*, 2019.
- [31] Pritam Sarkar and Ali Etemad. Self-supervised learning for ecg-based emotion recognition. In *ICASSP*, pages 3217–3221, 2020.
- [32] Joseph Y Cheng, Hanlin Goh, Kaan Dogrusoz, Oncel Tuzel, and Erdrin Azemi. Subject-aware contrastive learning for biosignals. *arXiv preprint arXiv:2007.04871*, 2020.
- [33] Sriram Ravula, Georgios Smyrnis, Matt Jordan, and Alexandros G Dimakis. Inverse problems leveraging pre-trained contrastive representations. *NeurIPS*, 34:8753–8765, 2021.
- [34] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, pages 1597–1607, 2020.
- [35] Zhigang Dai, Bolun Cai, Yugeng Lin, and Junying Chen. Up-detr: Unsupervised pre-training for object detection with transformers. In *CVPR*, pages 1601–1610, 2021.
- [36] Hsin-Ying Lee, Jia-Bin Huang, Maneesh Singh, and Ming-Hsuan Yang. Unsupervised representation learning by sorting sequences. In *Proceedings of the IEEE international conference on computer vision*, pages 667–676, 2017.
- [37] Mathilde Caron, Piotr Bojanowski, Julien Mairal, and Armand Joulin. Unsupervised pre-training of image features on non-curated data. In *ICCV*, pages 2959–2968, 2019.
- [38] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.
- [39] Neo Wu, Bradley Green, Xue Ben, and Shawn O’Banion. Deep transformer models for time series forecasting: The influenza prevalence case. In *arXiv:2001.08317*, 2020.
- [40] Chi Ian Tang, Ignacio Perez-Pozuelo, Dimitris Spathis, and Cecilia Mascolo. Exploring contrastive learning in human activity recognition for healthcare. *arXiv preprint arXiv:2011.11542*, 2020.
- [41] Dani Kiyasseh, Tingting Zhu, and David A Clifton. Clocs: Contrastive learning of cardiac signals across space, time, and patients. In *ICML*, pages 5606–5615, 2021.

- [42] David Berthelot, Rebecca Roelofs, Kihyuk Sohn, Nicholas Carlini, and Alex Kurakin. Adamatch: A unified approach to semi-supervised learning and domain adaptation. ICLR, 2022.
- [43] Guoqiang Wei, Cuiling Lan, Wenjun Zeng, Zhizheng Zhang, and Zhibo Chen. Toalign: Task-oriented alignment for unsupervised domain adaptation. NeurIPS, 34:13834–13846, 2021.
- [44] Tongkun Xu, Weihua Chen, Pichao Wang, Fan Wang, Hao Li, and Rong Jin. Cdtrans: Cross-domain transformer for unsupervised domain adaptation. ICLR, 2022.
- [45] Bernd Illing, Jean Ventura, Guillaume Bellec, and Wulfram Gerstner. Local plasticity rules can learn deep representations using self-supervised contrastive predictions. NeurIPS, 34:30365–30379, 2021.
- [46] Sana Tonekaboni, Danny Eytan, and Anna Goldenberg. Unsupervised representation learning for time series with temporal neighborhood coding. In ICLR, 2021.
- [47] Zhihan Yue, Yujing Wang, Juanyong Duan, Tianmeng Yang, Congrui Huang, Yunhai Tong, and Bixiong Xu. Ts2vec: Towards universal representation of time series. In AAAI, volume 36, pages 8980–8987, 2022.
- [48] 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.
- [49] Gerald Woo, Chenghao Liu, Doyen Sahoo, Akshat Kumar, and Steven Hoi. CoST: Contrastive learning of disentangled seasonal-trend representations for time series forecasting. In ICLR, 2022.
- [50] Ling Yang and Shenda Hong. Unsupervised time-series representation learning with iterative bilinear temporal-spectral fusion. In ICML, pages 25038–25054. PMLR, 2022.
- [51] Rob J Hyndman and George Athanasopoulos. Forecasting: principles and practice. OTexts, 2018.
- [52] Ronald Newbold Bracewell and Ronald N Bracewell. The Fourier transform and its applications, volume 31999. McGraw-hill New York, 1986.
- [53] Leon Cohen. Time-frequency analysis, volume 778. Prentice hall New Jersey, 1995.
- [54] Henri J Nussbaumer. The fast fourier transform. In Fast Fourier Transform and Convolution Algorithms, pages 80–111. Springer, 1981.
- [55] Patrick Flandrin. Time-frequency/time-scale analysis. Academic press, 1998.
- [56] Antonia Papandreou-Suppappola. Applications in time-frequency signal processing. CRC press, 2018.
- [57] Ryan Soklaski, Michael Yee, and Theodoros Tsiligkaridis. Fourier-based augmentations for improved robustness and uncertainty calibration. NeurIPS’W, 2021.
- [58] Ashish Jaiswal, Ashwin Ramesh Babu, Mohammad Zaki Zadeh, Debapriya Banerjee, and Fillia Makedon. A survey on contrastive self-supervised learning. Technologies, 9(1):2, 2020.
- [59] Elad Hoffer and Nir Ailon. Deep metric learning using triplet network. In International workshop on similarity-based pattern recognition, pages 84–92. Springer, 2015.
- [60] Vassileios Balntas, Edgar Riba, Daniel Ponsa, and Krystian Mikolajczyk. Learning local feature descriptors with triplets and shallow convolutional neural networks. In Bmvc, volume 1, page 3, 2016.
- [61] Bob Kemp, Aeilko H Zwinderman, Bert Tuk, Hilbert AC Kamphuisen, and Josefien JL Obery. Analysis of a sleep-dependent neuronal feedback loop: the slow-wave microcontinuity of the eeg. IEEE Transactions on Biomedical Engineering, 47(9):1185–1194, 2000.
- [62] Ralph G Andrzejak, Klaus Lehnertz, Florian Mormann, Christoph Rieke, Peter David, and Christian E Elger. Indications of nonlinear deterministic and finite-dimensional structures in time series of brain electrical activity: Dependence on recording region and brain state. Physical Review E, 64(6):061907, 2001.
- [63] Christian Lessmeier, James Kuria Kimotho, Detmar Zimmer, and Walter Sestro. Condition monitoring of bearing damage in electromechanical drive systems by using motor current signals of electric motors: A benchmark data set for data-driven classification. In PHM Society European Conference, volume 3, 2016.

- [64] Davide Anguita, Alessandro Ghio, Luca Oneto, Xavier Parra Perez, and Jorge Luis Reyes Ortiz. A public domain dataset for human activity recognition using smartphones. In *ESANN*, pages 437–442, 2013.
- [65] Jiayang Liu, Lin Zhong, Jehan Wickramasuriya, and Venu Vasudevan. uwave: Accelerometer-based personalized gesture recognition and its applications. *Pervasive and Mobile Computing*, 5(6):657–675, 2009.
- [66] Ary L Goldberger, Luis AN Amaral, Leon Glass, Jeffrey M Hausdorff, Plamen Ch Ivanov, Roger G Mark, Joseph E Mietus, George B Moody, Chung-Kang Peng, and H Eugene Stanley. Physiobank, physiotoolkit, and physionet: components of a new research resource for complex physiologic signals. *circulation*, 101(23):e215–e220, 2000.
- [67] Amrutha Ramanathan and James McDermott. Fall detection with accelerometer data using residual networks adapted to multi-variate time series classification. In *IJCNN*, pages 1–8, 2021.
- [68] George Zerveas, Srideepika Jayaraman, Dhaval Patel, Anuradha Bhamidipaty, and Carsten Eickhoff. A transformer-based framework for multivariate time series representation learning. In *KDD*, pages 2114–2124, 2021.
- [69] Xiang Zhang and Lina Yao. *Deep Learning for EEG-Based Brain-Computer Interfaces: Representations, Algorithms and Applications*. World Scientific, 2021.
- [70] Julien Audibert, Pietro Michiardi, Frédéric Guyard, Sébastien Marti, and Maria A Zuluaga. Usad: Unsupervised anomaly detection on multivariate time series. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 3395–3404, 2020.
- [71] Xiang Zhang, Marko Zeman, Theodoros Tsiligkaridis, and Marinka Zitnik. Graph-guided network for irregularly sampled multivariate time series. In *ICLR*, 2022.
- [72] Max Horn, Michael Moor, Christian Bock, Bastian Rieck, and Karsten Borgwardt. Set functions for time series. In *ICML*, pages 4353–4363, 2020.

Broader Impacts

Our approach for self-supervised pre-training improves classification performance on target datasets in different application scenarios. The recognition of time-frequency consistency as a universal property specific to time series data is a weak assumption that enables effective, task- and domain-agnostic transfer learning. We believe our work will inspire the research community to uncover other universal properties for transfer learning. We also hope our work will also attract more researchers to the more general problem of time series representation learning which is still underappreciated relative to problems from CV and NLP fields.

On the society level, our work, along the line of transfer learning, can facilitate more efficient use of time series data in various settings. For example, in medical settings, some diseases of clinical interest may have very small labelled dataset. In this case, unlabelled data from patients of different diseases but with similar underlying physiological conditions can be used to pre-train the model. However, practitioners need to be aware of the limitations of the model, including that it may make biased predictions. Specifically, bias may exist in the source dataset used for pre-training due to an imbalance of samples from subjects of different demographic attributes. Also, the standardized medical protocols for collecting these datasets might be unsuitable for subjects with certain physiological attributes, creating unforeseen bias that may be transferred to fine-tuning.

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

Checklist

1. For all authors...
 - (a) Do the main claims made in the abstract and introduction accurately reflect the paper’s contributions and scope? [\[Yes\]](#) In abstract and introduction, we claim that TF-C is a generalizable property of time series that can support pre-training, which is well-justified in Sec. 3 and experimentally demonstrated in Sec. 5 (our model consistently performs comparatively to or above baseline methods).
 - (b) Did you describe the limitations of your work? [\[Yes\]](#) See Section 6.
 - (c) Did you discuss any potential negative societal impacts of your work? [\[Yes\]](#) See Broader Impact on Page 10.
 - (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [\[Yes\]](#)
2. If you are including theoretical results...
 - (a) Did you state the full set of assumptions of all theoretical results? [\[N/A\]](#)
 - (b) Did you include complete proofs of all theoretical results? [\[N/A\]](#)
3. If you ran experiments...
 - (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [\[Yes\]](#) Yes, we include an anonymous link (see Abstract) that provides the source codes with all implementation details, implementation of baselines, and eight datasets. The link will be updated to a non-anonymous link after acceptance.
 - (b) Did you specify all the training details (e.g., data splits, hyper-parameters, how they were chosen)? [\[Yes\]](#) See implementation details in Sec. 5. See Appendix E for baseline architectures and hyper-parameter settings. More details can be found in the included URL.
 - (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [\[Yes\]](#) We run experiments for 5 times and report the average value with standard deviation. See Table 1, Tables 4-6, and Table 2.
 - (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [\[Yes\]](#) See Appendix E.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...

- (a) If your work uses existing assets, did you cite the creators? **[Yes]** We used eight existing datasets and 6 state-of-the-art baselines in contrastive learning and pre-training for time series. We cited the creators for every exist asset we used. See Sec. 5.
 - (b) Did you mention the license of the assets? **[Yes]** All dataset licenses are mentioned in the Appendix D.
 - (c) Did you include any new assets either in the supplemental material or as a URL? **[Yes]** See the anonymous link in Abstract.
 - (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? **[No]** All data we use is freely available for download, without any requirement to re-contact the data curator.
 - (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? **[No]** Our datasets are public, well-established, and do not contain PII or offensive content
5. If you used crowdsourcing or conducted research with human subjects...
- (a) Did you include the full text of instructions given to participants and screenshots, if applicable? **[N/A]**
 - (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? **[N/A]**
 - (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? **[N/A]**