Enhancing Representation Learning for Periodic Time Series with Floss: A Frequency Domain Regularization Approach

Chunwei Yang
Sichuan University
Chengdu, China
ycwcw123@gmail.com

Xiaoxu Chen
McGill University
Montreal, Canada
xiaoxu.chen@mail.mcgill.ca

Lijun Sun
McGill University
Montreal, Canada
lijun.sun@mcgill.ca

Hongyu Yang
Sichuan University
Chengdu, China
yanghongyu@scu.edu.cn

Yuankai Wu
Sichuan University
Chengdu, China
wuyk0@scu.edu.cn

ABSTRACT

Time series analysis is a fundamental task in various application domains, and deep learning approaches have demonstrated remarkable performance in this area. However, many real-world time series data exhibit significant periodic or quasi-periodic dynamics that are often not adequately captured by existing deep learning-based solutions. This results in an incomplete representation of the underlying dynamic behaviors of interest. To address this gap, we propose an unsupervised method called Floss that automatically regularizes learned representations in the frequency domain. The Floss method first automatically detects major periodicities from the time series. It then employs periodic shift and spectral density similarity measures to learn meaningful representations with periodic consistency. In addition, Floss can be easily incorporated into both supervised, semi-supervised, and unsupervised learning frameworks. We conduct extensive experiments on common time series classification, forecasting, and anomaly detection tasks to demonstrate the effectiveness of Floss. We incorporate Floss into several representative deep learning solutions to justify our design choices and demonstrate that it is capable of automatically discovering periodic dynamics and improving state-of-the-art deep learning models.

:
Chunwei Yang, Xiaoxu Chen, Lijun Sun, Hongyu Yang, and Yuankai Wu. Enhancing Representation Learning for Periodic Time Series with Floss: A Frequency Domain Regularization Approach. 14(1): XXX-XXX, 2020.
doi:XX.XX/XXX.XX

The source code, data, and/or other artifacts have been made available at https://github.com/AgustDD/Floss.

1 INTRODUCTION

We are witnessing continued developments in sensor technologies, where sensors produce multivariate time series. These advances have paved the way for the critical role of time series analysis in various scientific and engineering fields. In the realm of energy management, time series analysis enables accurate load forecasting, facilitating efficient resource allocation and optimal energy utilization [7, 15]. Within transportation engineering, time series analysis plays a pivotal role in predicting traffic flows and optimizing transportation systems [19, 43, 44]. Moreover, in financial markets, time series analysis is of utmost importance. It allows for the modeling of asset prices, enables volatility forecasting, and assists in developing effective risk management strategies [34]. The application of time series analysis in healthcare proves invaluable as well, aiding in patient monitoring, disease surveillance, and the prediction of health outcomes [20, 29].

The widespread adoption of deep neural networks in time series analysis has brought about significant advancements in recent years [28]. These models have demonstrated their efficacy in capturing complex temporal patterns by leveraging supervised or unsupervised training approaches. Through proper training method, neural networks acquire robust temporal representations that are well-suited for various tasks within time series analysis [47]. One crucial task where neural networks excel is forecasting, where they leverage their learned temporal representations to make accurate predictions about future values [42]. Additionally, neural networks have shown promising results in anomaly detection within time series data [26].

The quest for a universal representation of time series data has sparked significant interest in deep representation learning strategies, including contrastive learning [4, 14]. These strategies aim to extract powerful representations from the hidden layers of deep neural networks, capturing the intrinsic features embedded within time series data. The value of such representations extends to various downstream tasks, including time series anomaly detection, forecasting, and classification. Researchers have explored specific invariances within time series data to enhance deep representation learning frameworks. For instance, Franceschi et al. [10] encouraged representations that closely resemble sampled subseries, while Tonekaboni et al. [32] enforced smoothness between adjacent time windows. Eldele et al. [9] proposed a model that learns scale and permutation-invariant representations. Yue et al. [47] introduced TS2Vec, a contrastive learning framework that captures contextual invariances at multiple resolutions within time series. Despite the progress made, existing methods often borrow ideas directly from contrastive learning methods in computer vision and natural language processing domains. However, unlike images that typically possess recognizable features, time series data often exhibit underlying patterns that are not easily explainable. Applying assumptions

Corresponding author (Yuankai Wu, wuyk0@scu.edu.cn)

borrowed from other domains without careful consideration may result in unsuccessful representation learning for time series data.

The temporal dynamics of real-world processes often exhibit recurring cycles and significant periodicity, which are fundamental characteristics of time series data [11]. This inherent property becomes particularly evident in time series associated with human behavior, where prominent daily and weekly patterns emerge. Recognizing the importance of capturing and leveraging periodicity, the exploration of representation learning methods that effectively capture the underlying periodic invariance holds substantial promise in time series analysis. One classical approach to detecting periodicity in traditional time series analysis is the employment of frequency domain methods, which enable the identification of periodic patterns by transforming time series into the frequency domain [21]. The discrete Fourier transform (DFT), for instance, facilitates the conversion of time series from the time domain to the frequency domain, yielding the periodogram that encodes the strength at different frequencies. Similarly, other transformations, such as the discrete cosine transform and wavelet transform, can also identify periodicity and enhance supervised learning in time series analysis [38]. These studies provide compelling evidence that frequency domain information harbors valuable insights for analyzing periodic time series data.

In fact, frequency-domain information has been widely leveraged in deep learning architectures for modeling time series data. Zhou et al. [53] proposed the use of Transformers operating in the frequency domain, enabling the capture of global properties within time series data. Woo et al. [40] introduced ETSTransformer, which utilizes Fourier bases in the frequency domain to extract dominant seasonal patterns from time series. Liu et al. [22] devised a tree-structured network that iteratively decomposes input signals into various frequency subbands. Zhang et al. [50] decomposed time series into seasonal and trend components, employing Fourier attention for prediction. Wu et al. [41] employed Fourier transformation to disentangle original temporal variations into intraperiod and interperiod variations, capturing their dependencies using 2D convolutional operations. Notably, recent efforts have focused on regulating frequency-domain representations through unsupervised learning approaches. Zhang et al. [51] directly applied contrastive learning to the frequency transformation of raw signals, embedding the time-based neighborhood of an example close to its frequency-based neighborhood. Similar ideas were explored in CoST [39] and BTSF [46]. However, while these approaches leverage unsupervised learning and contrastive learning, none of them are specifically designed to capture periodic dynamics in time series data.

In our pursuit of capturing periodic dynamics by time series representations, we propose a novel approach that leverages the principles of contrastive learning [4, 17]. Contrastive learning operates on the basis of two key elements: (i) a contrastive loss that compares features and (ii) a set of transformations that encode the desired invariances. Building upon this framework, we introduce a simple yet effective combination of loss function and transformation named Floss, which can be seamlessly integrated into unsupervised and semi-supervised learning methods specifically designed for periodic time series analysis. Our approach centers on the hypothesis that **the spectral density of the learned representation remains invariant under periodic transformations**. To realize this, our framework incorporates straightforward and efficient data augmentations that can accommodate various periodic time series with specified periodicities. Initially, we employ frequency domain transformation to automatically detect the dominant periodicity and create a periodic view of the target time series by introducing random periodic shifts in the temporal dimension. Subsequently, a time series encoder is employed to learn a periodic invariant representation. Importantly, this encoder can be seamlessly integrated into any existing deep learning framework, thereby ensuring compatibility and flexibility in its application. Finally, we design a novel task that enforces the similarity of spectral densities between the target time series and its periodic views. To mitigate the influence of high-frequency noise, we employ a hierarchical approach to measure the similarity of spectral densities between the representations. The intuition of our work is illustrated in Figure 1.

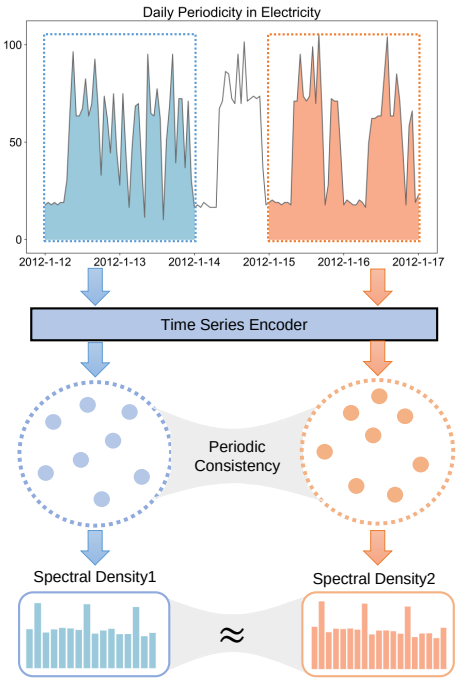


Figure 1: The framework of the paper: The time series shown in the figure exhibits strong daily periodicity. After detecting this periodicity, we aim to make the spectral densities of the representations of two time series segments, which differ by several number of periodicities, as similar as possible.

To the best of our knowledge, this study represents the first systematic investigation into the learning of representations for periodic or quasi-periodic time series by examining the invariance of spectral density. Specifically, our Floss can be seamlessly integrated into current supervised and unsupervised frameworks. The outcomes obtained from tasks such as time series classification, forecasting, and anomaly detection confirm the ability of Floss to capture and encode periodic invariances in time series, resulting in a notable enhancement of task performance.

The paper is organized as follows. Section 2 introduces the necessary concepts for understanding the Floss system. In Section 3, we provide a comprehensive description of our Floss framework. Furthermore, Section 4 showcases the results of our forecasting, classification, and anomaly detection experiments using the Floss-enhanced models on extensive benchmarking datasets. In addtion, in-depth analysis and ablation studies are also provided in Section 4. Finally, Section 5 offers concluding remarks and summarizes our work.

2 PRELIMINARIES

Periodic time series: Given a data set of periodic time series, denoted $\mathcal{X} \in \mathbb{R}^{N \times T \times F}$, where N represents the number of time series and T and F indicate the size of the time window and feature dimension, respectively, we assume that these time series exhibit periodic behavior. Moreover, it is important to note that the periodicities may vary within the sampled time ranges. To further clarify, let's define $[t_1, t_2] = \{t_1, t_1 + 1, \ldots, t_2 - 1, t_2\}$. We use the notation $\mathcal{X}_{[t_1,t_2]} \in \mathbb{R}^{N \times (t_2 - t_1 + 1) \times F}$ to represent the time series sampled from t_1 to t_2.

To illustrate, let's consider the scenario where $\mathcal{X}$ represents traffic time series collected from N traffic sensors in a road network. If we sample the data over a period corresponding to a single day for $\mathcal{X}_{[t_1,t_2]}$, it becomes apparent that the dominant periodicity is approximately 6 hours, as traffic data typically exhibits morning and evening peaks. Conversely, if we sample the data over several days for $\mathcal{X}_{[t_1,t_2]}$, the prominent period would be one day. Furthermore, it is worth noting that time series can exhibit multiple periodicities. For instance, in the traffic example, there could be periodicities of 6 hours and 1 day. We introduce the notation $p_{[t_1,t_2]} \in \mathbb{R}$ to denote the prominent periodicity of time series $\mathcal{X}_{[t_1,t_2]}$.

Time series representation: For a given $\mathcal{X}_{[t_1,t_2]}$, a representation model $\mathcal{G}(\cdot;\theta)$ parameterized by θ generates a representation tensor $\mathcal{Y}_{[t_1,t_2]} = \mathcal{G}\left(\mathcal{X}_{[t_1,t_2]};\theta\right)$. Here, $\mathcal{Y}_{[t_1,t_2]} \in \mathbb{R}^{N' \times (t_2 - t_1 + 1) \times F'}$, where N' and F' indicate the dimensions of the modified time series count and the representation feature, respectively. It is important to note that the value of N' varies depending on the choice of $\mathcal{G}$. If we aim to generate an overall representation encompassing all time series, then $N' = 1$. On the other hand, if the goal is to produce a representation for each individual time series, then $N' = N$.

Power Spectral Density: In signal processing, the power spectral density provides information about the expected signal power at different frequencies of the signal. For example, the periodogram is a measure of spectral density in the Fourier domain. Denoting the discrete Fourier transform as $\mathcal{DFT}(\cdot)$, the periodogram $\mathbf{\Phi}(\cdot)$ is computed as:

$$\begin{aligned} \mathcal{DFT}\left(w_j\right) &= \frac{1}{\sqrt{n}} \sum_{t=1}^{n} x_t e^{-2\pi i w_j t}, \\ \mathbf{\Phi}\left(w_j\right) &= \mathrm{Re}\left(\mathcal{DFT}\left(w_j\right)\right)^2 + \mathrm{Im}\left(\mathcal{DFT}\left(w_j\right)\right)^2, \end{aligned} \tag{1}$$

where x_t denotes the time series value at time point t, $\mathrm{Re}(\cdot)$ and $\mathrm{Im}(\cdot)$ denote the real and imaginary parts, respectively. Each element of the periodogram represents the power at frequency w_j, or equivalently, at period $1/w_j$. It is important to note that other transformations, such as discrete cosine transform (DCT) and wavelet transform (DWT), can also be used to calculate the spectral density. If we employ the DCT, the transformation is given by:

$$\begin{aligned} \mathcal{DCT}(w_j) &= \left(\frac{n}{2}\right)^{-1/2} \sum_{t=1}^{n} \wedge(t) x_t \cos\left(\frac{\pi w_j}{2n}(2t-1)\right), \\ \wedge(t) &= \begin{cases} \frac{1}{\sqrt{2}} & \text{if} \quad t = 1 \\ 1 & \text{otherwise} \end{cases} \\ \mathbf{\Phi}\left(w_j\right) &= \left|\mathcal{DCT}\left(w_j\right)\right|. \end{aligned} \tag{2}$$

3 METHOD

In this section, we present the proposed frequency domain loss (Floss) for periodic time series and provide implementation details. Floss is an novel framework that aims to capture the inherent periodic invariance of time series in its learned representations. To accomplish this, the framework incorporates two key steps: a periodicity detection module for generating periodic views and a novel objective that compares the spectral densities of these representations (Figure 2). By doing so, the learned representations are equipped with an awareness of the underlying periodic nature of time series.

3.1 Periodic Detection and Augmentation

Assuming the existence of multiple periodicities within each temporal sampled time series $\mathcal{X}_{[t_1,t_2]} \in \mathbb{R}^{N \times (t_2 - t_1 + 1) \times F}$, our study focuses on a wide time range $[t_1, t_2]$ to encompass diverse and significant periodic patterns in the data. In order to create periodic transformation, it is necessary to first identify the underlying periods. This is achieved by calculating the average spectral density using the following procedure:

$$\begin{aligned} \hat{\mathbf{\Phi}} &= \frac{1}{NF} \sum_{n=1}^{N} \sum_{f=1}^{F} \mathbf{\Phi}_{\mathbf{n},\mathbf{f}}, \\ \hat{w} &= \arg\max\left(\hat{\mathbf{\Phi}}\right), \\ \hat{p}_{[t_1,t_2]} &= \frac{(t_2 - t_1 + 1)}{\hat{w}}. \end{aligned} \tag{3}$$

Here, $\mathbf{\Phi}_{\mathbf{n},\mathbf{f}}$ represents the estimated periodogram of the f-th feature of the n-th time series. The symbol $\hat{\mathbf{\Phi}} \in \mathbb{R}^{t_2 - t_1 + 1}$ denotes the average periodogram across features. It is important to note that the j-th value $\mathbf{\Phi}(w_j)$ signifies the intensity of the frequency-j periodic basis function, which is associated with the period length $\frac{(t_2 - t_1 + 1)}{w_j}$. Furthermore, we examine the maximum periodicity $\hat{p}_{[t_1,t_2]}$ discovered through the periodogram, which corresponds to the highest value observed in $\hat{\mathbf{\Phi}}$.

Although the periodogram is extensively employed for spectral analysis and capturing periodic dynamics, its efficacy can be suboptimal under certain circumstances. Notably, high levels of noise can obfuscate the periodic signals, resulting in inaccurate or potentially deceptive outcomes [33]. Additionally, the periodogram may encounter challenges when faced with complex spectral shapes or irregular patterns, impeding its ability to precisely capture and characterize the underlying periodic dynamics [37].

In our approach, we compute a periodogram for each sampled batch, which essentially involves random sampling over the time

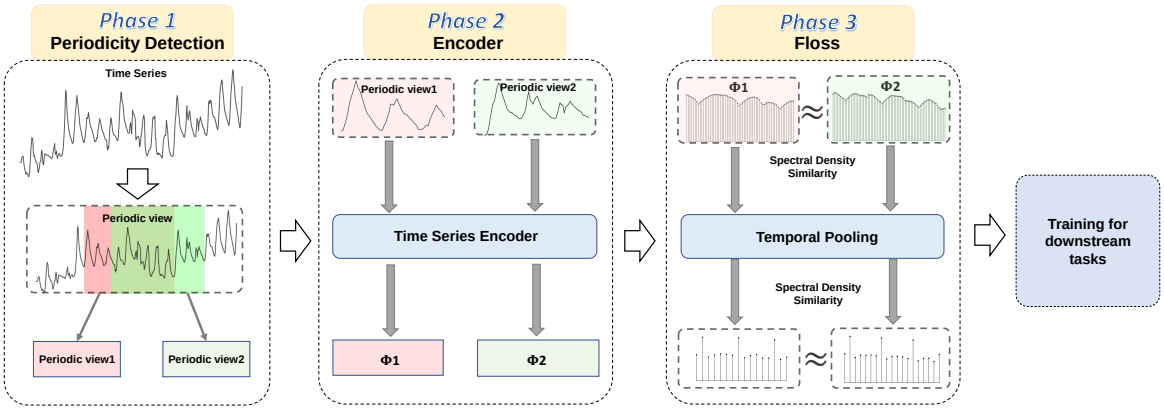


Figure 2: Our framework comprises three critical steps: (1) Periodicity Detection: We automatically detect periodicity patterns from the input time series samples and utilize the detected periodicity to create two views of the input time series. (2) Frequency Domain Similarity Learning: The two periodic views are processed through their respective time series encoders, generating two representations. (3) The Floss algorithm hierarchically calculates the similarities between the spectral densities of the two representations using temporal pooling. The pre-trained encoder can then be directly applied to downstream tasks.

domain during the training period. We posit that the potential inaccuracies associated with the periodogram can be mitigated by employing this temporal sampling approach. By performing random sampling over a wide time range, we increase the number of samples, thereby enhancing the statistical consistency of the estimated periodogram. This approach is supported by empirical validation. For instance, in the field of signal processing, random sampling followed by periodogram analysis has proven effective in identifying periodic signals [31]. Similarly, in astronomy, this approach has been successfully utilized for periodogram analysis [35].

After obtaining the estimated $\hat{p}_{[t_1,t_2]}$ for $\mathcal{X}_{[t_1,t_2]}$, we shift the data along the time axis to exploit the periodic dynamics. We implement this concept through random periodic shifts. In a formal sense, we consider the periodic view of $\mathcal{X}_{[t_1,t_2]}$ as $\mathcal{X}_{[\hat{t}_1,\hat{t}_2]}$, where $\hat{t}_1$ and $\hat{t}_2$ are $t_1 + a\hat{p}_{[t_1,t_2]}$ and $t_2 + a\hat{p}_{[t_1,t_2]}$, a is a random integer,

3.2 Hierarchical Frequency-Domain Loss

Given an encoder $\mathcal{G}(\cdot;\theta)$ parameterized by θ, along with the original view $\mathcal{X}_{[t_1,t_2]}$ and its periodic view $\mathcal{X}_{[\hat{t}_1,\hat{t}_2]}$, our objective is to minimize the difference in power spectral density between the two representations. Let $\mathcal{Y} = \mathcal{G}\left(\mathcal{X}_{[t_1,t_2]};\theta\right)$ and $\hat{\mathcal{Y}} = \mathcal{G}\left(\mathcal{X}_{[\hat{t}_1,\hat{t}_2]};\theta\right)$. Let $\Phi_{\mathcal{Y}}$ and $\Phi_{\hat{\mathcal{Y}}}$ represent the estimated periodograms of $\mathcal{Y}$ and $\hat{\mathcal{Y}}$ respectively. The loss function for achieving periodic invariance can be defined as follows:

$$\mathcal{L}_f = \frac{1}{N'F'}\|\Phi_{\mathcal{Y}} - \Phi_{\hat{\mathcal{Y}}}\|_{l1}, \tag{4}$$

where N' and F' denote the projected time series and the number of features in $\mathcal{Y}$ and $\hat{\mathcal{Y}}$ respectively.

By minimizing the loss function defined in Equation (4), we can reap two distinct advantages of preserving periodic invariance. Firstly, it ensures that the representations of the original view and its periodic counterpart exhibit similarity within a specific domain. Secondly, it enables the identification of similar periodic patterns from the representations of both the original view and its periodic view.

However, retaining all frequency components, as in Equation (4), may lead to subpar representations, as many high-frequency fluctuations in time series can be attributed to noisy inputs. Conversely, exclusively preserving low-frequency components might not be suitable for time series modeling, as certain shifts in trends within the time series carry significant meaning. To better capture information from all frequency components, we propose a hierarchical frequency loss, which compels the encoder to learn representations at multiple scales. Our approach involves hierarchically applying temporal max pooling to the learned features $\mathcal{Y}$ and $\hat{\mathcal{Y}}$, followed by computing their periodic invariance loss. The algorithmic steps for this calculation are outlined in Algorithm 1. Temporal max pooling selects the most prominent element within a given region of the representation, thereby yielding an output that retains the salient features while minimizing noise interference. Furthermore, the temporal pooling operation reduces the temporal dimensionality of the hidden representation. Consequently, the corresponding frequency component of the hidden representation decreases after max pooling, enabling greater emphasis on the low-frequency component. This strategy is reasonable, considering our objective is to encode periodic invariance, which primarily resides within the low-frequency domain.

In Algorithm 1, the parameter τ plays a crucial role in controlling the weighting of high-frequency components in the context of max pooling. A larger value of τ assigns greater importance to the high-frequency parts. For instance, setting τ to match the temporal length of the feature would effectively equate it to directly comparing the spectral densities of the two features. It is noteworthy that in our experiment, we discovered certain datasets where employing non-hierarchical Floss and directly comparing spectral densities

Algorithm 1 Calculating the hierarchical frequency loss

Input:$\mathcal{Y}$, $\hat{\mathcal{Y}}$, a spectral density measure Φ
Parameter: Pooling scale τ
Output: Hierarchical Loss $\mathcal{L}_{hier}$

$\mathcal{L}_{hier} \leftarrow \mathcal{L}_f(\mathcal{Y}, \hat{\mathcal{Y}}, \Phi(\cdot))$;
$d \leftarrow 1$
while length$(\mathcal{Y}) > 1$ **do**
  $\mathcal{Y} \leftarrow \text{maxpool1d}(\mathcal{Y}, \tau)$;
  $\hat{\mathcal{Y}} \leftarrow \text{maxpool1d}\left(\hat{\mathcal{Y}}, \tau\right)$;
  $\mathcal{L}_{hier} \leftarrow \mathcal{L}_{hier} + \mathcal{L}_f(\mathcal{Y}, \hat{\mathcal{Y}}, \Phi(\cdot))$;
  $d \leftarrow d + 1$;
end while
$\mathcal{L}_{hier} \leftarrow \mathcal{L}_{hier}/d$;
return $\mathcal{L}_{hier}$.

produced superior outcomes. Subsequent analyses will delve deeper into this particular aspect.

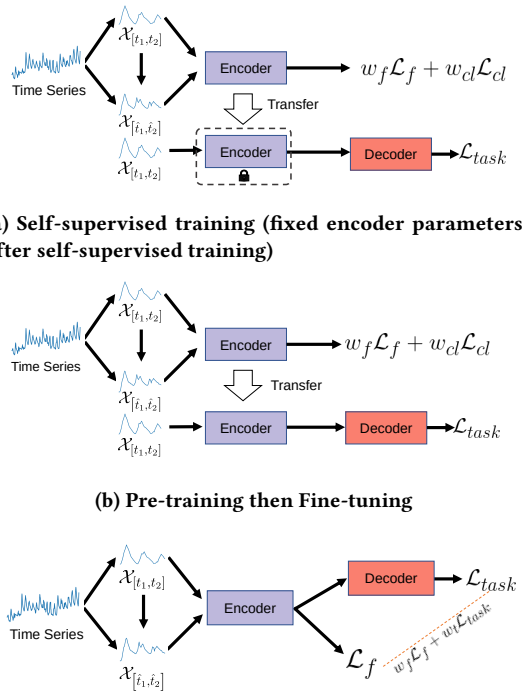
3.3 Training Schemes Under Different Settings

The Frequency-domain loss (Floss) function, which is proposed in Section 3.2, can be readily employed in both supervised and unsupervised learning settings. This section explores the integration of Floss into unsupervised, semi-supervised, and supervised time series analysis. We summarize the training strategy of different schemes in Figure 3

1) Self-supervised training: In the pretraining phase, only the unlabeled time series $\mathcal{X} \in \mathbb{R}^{N \times T \times F}$ are available. First, we randomly sample the original view $\mathcal{X}_{[t_1,t_2]}$ and its periodic view $\mathcal{X}_{[\hat{t}_1,\hat{t}_2]}$ from $\mathcal{X}$, considering periodic shifts. To make Floss compatible with other self-supervised learning schemes, we can apply augmentation techniques such as timestamp masking and random cropping [47] to $\mathcal{X}_{[t_1,t_2]}$ and $\mathcal{X}_{[\hat{t}_1,\hat{t}_2]}$. Subsequently, we pass the original and transformed inputs through an encoder $G(\cdot;\theta)$. The Floss is computed using the representations $G\left(\mathcal{X}_{[t_1,t_2]};\theta\right)$ and $G\left(\mathcal{X}_{[\hat{t}_1,\hat{t}_2]};\theta\right)$. The Floss $\mathcal{L}_f$ can be combined with other self-supervised loss functions using a weighted combination $\mathcal{L}_f$ and other contrastive learning loss $\mathcal{L}_{cl}$ to train the encoder $G(\cdot;\theta)$. During this stage, the downstream tasks are assumed to be unknown. Finally, we follow the same protocol as [10], where a decoder is trained on top of the representations $G\left(\mathcal{X}_{[\hat{t}_1,\hat{t}_2]};\theta\right)$ to handle the downstream tasks. It is important to note that the parameters θ of the encoder remain fixed during the final training phase.

2) Pre-training then Fine-tuning: The procedure for pretraining in the semi-supervised setting is similar to that of the unsupervised setting. However, during the fine-tuning stage, the optimized model parameters θ of $G(\cdot;\theta)$ are further fine-tuned to transition from $G(\cdot;\theta)$ to $G(\cdot;\phi)$ using the downstream tasks.

3) Joint training under supervised learning setting: In the joint training approach, both the encoder and decoder are trained simultaneously. In this scenario, the Floss serves as an auxiliary regularization term during training, providing additional self-supervision signals that contribute to enhancing generalization. Specifically, in this setting, both the unlabeled time series $\mathcal{X} \in \mathbb{R}^{N \times T \times F}$ and their corresponding labels $\mathcal{D}$ are available during the training phase. The encoder $G(\cdot;\theta)$ is trained using a weighted combination of the Floss $\mathcal{L}_f$ and the supervised loss $\mathcal{L}_{task}$.



(a) Self-supervised training (fixed encoder parameters after self-supervised training)

(b) Pre-training then Fine-tuning

(c) Joint Training

Figure 3: Illustration of different training schemes.

4 EXPERIMENTS

In this section, we assess the effectiveness of Floss in periodic time series forecasting, classification, and anomaly detection. Our primary objective in this study is to determine whether incorporating Floss can enhance the performance of current supervised and unsupervised representation learning frameworks.

4.1 Multivariate time series forecasting

4.1.1 Existing Algorithms. We consider three representative multivariate time series forecasting models: 1). TS2Vec [47]: This is a purely unsupervised learning model. TS2Vec employs contrastive learning in a hierarchical manner on augmented views. Its encoder is based on a lightweight temporal convolutional network. After training, the encoder remains fixed, and ridge regression is used for the forecasting task. To integrate Floss into TS2Vec, we adapt the augmentation strategy of TS2Vec to incorporate periodic shifts. We randomly sample two segments $[t_1 - j_1, t_2 + k_1]$ and $[t_1 + a\hat{p}[t_1,t_2] - j_2, t_2 + a\hat{p}[t_1,t_2] + k_2]$, where a, j_1, j_2, k_1, and k_2 are random integers. We also apply TS2Vec's timestamp mask strategy to the time series segments. Then, we train our model using a weighted sum of the frequency loss and contrastive loss from TS2Vec on the representations of the segments $[t_1, t_2]$ and

$[t_1 + a\hat{p}[t_1, t_2], t_2 + a\hat{p}[t_1, t_2]]$. The estimated periodicity and frequency loss are computed using discrete cosine transformation (DCT). 2) PatchTST [24]: This model employs a vision Transformer-style architecture for multivariate time series forecasting and utilizes pre-training and fine-tuning techniques for training. In the self-learning phase, the model is trained to reconstruct masked time series patches. After self-training, the transformer is fine-tuned for downstream multivariate forecasting tasks. In our approach, Floss collaborates with the reconstruction loss in a weighted sum fashion during the self-training phase. 3) Informer [52]: This transformer model is a milestone in time series forecasting and is trained using a purely supervised learning approach. We incorporate Floss to regularize its hidden representation, specifically the layer before the final layer. The model is trained by combining the forecasting loss and the proposed frequency loss using a weighted sum.

Not only do we choose models based on the paradigms of different training schemes, but the sizes of these three models are also representative. TS2Vec has a relatively small structure, Informer is of medium size, while PatchTST is a larger model.

4.1.2 Public Datasets. We assess the effectiveness of our proposed Floss by evaluating its performance on 8 widely-used datasets, namely Weather, Exchange, Electricity, ILI, and 4 ETT datasets (ETTh1, ETTh2, ETTm1, ETTm2). These datasets are commonly employed for benchmarking purposes and are publicly available on [48]. For the TS2Vec model, we allocated 60% of the data for training, 20% for validation, and 20% for testing. For the PatchTST and Informer models, we allocated 70% of the data for training, 10% for validation, and 20% for testing. The statistics of those datasets are summarized in Table 1.

4.1.3 Experimental Settings. Following previous works [48, 52, 53], we use Mean Squared Error (MSE) and Mean Absolute Error (MAE) as the core metrics to compare performance. All of the models follow the same experimental setup with a prediction length of $T \in \{24, 36, 48, 60\}$ for the ILI dataset and $T \in \{96, 192, 336, 720\}$ for other datasets, as mentioned in the original papers. For PatchTST and Informer, the lookback window is set to $L = 96$. We adhere to the standard protocol and split all datasets into training, validation, and test sets in chronological order using a ratio of 7:1:2. For TS2Vec, the lookback window is set equal to the prediction length T, and all datasets are split into training, validation, and test sets in the ratio of 6:2:2 (same as the original paper [47]).

The detailed hyper-parameter configurations of informer-Floss are set as follows: The batch size for all datasets is set to 32. Loss weights for different datasets are as follows: Weather (original forecasting loss weight= 0.3, Floss weight= 2), Exchange (original loss weight= 0.3, Floss weight= 0.7 for 96-step ahead prediction, Floss weight= 0.8 for all other prediction horizons), Electricity (original loss weight= 0.3, Floss weight= 2), ILI (original loss weight= 0.3, Floss weight= 0.5), ETTh1 (original loss weight= 0.3, Floss weight= 1), ETTh2 (original loss weight= 0.5, Floss weight= 8), ETTm1, and ETTm2 (original loss weight= 0.5, Floss weight= 8).

The detailed hyper-parameter configurations of TS2Vec-Floss are as follows: The batch size is set to 16, the Floss weight for the contrastive loss of TS2Vec is set to 1, and the loss for the contrastive loss of TS2Vec is assigned a value of 1. Similarly, the detailed hyper-parameter configurations of PatchTST-Floss are as follows: During pretraining, the reconstruction loss is set to 0.3, the Floss weight is set to 1, and the mask ratio is set to 0.4. Additionally, the batch size for Weather, Electricity, ETTh1, ETTh2, ETTm1, and ETTm2 datasets is set to 8, while for Exchange and ILI datasets, it is set to 16.

4.1.4 Experimental Results. Table 2 shows the multivariate long-term forecasting results. It should be noted that we reran the experiments for fair comparison; therefore, the performance of Informer, TS2Vec, and PatchTST is slightly better than what was reported in the original literature. We use bold text to highlight the improved performance and red color to indicate the average improvements. The key observations are as follows:

First, the inclusion of Floss enhances the overall performance of all three representative models. This demonstrates that Floss effectively utilizes informative features within the frequency domain, leading to improved forecasting performance.

Secondly, Floss performs remarkably well on the Electricity dataset, which includes the largest number (321) of time series in our experiments. Improvements are observed in all cases, indicating that Floss has the ability to encode shared frequency information from a large number of time series, thereby enhancing forecasting performance.

Thirdly, the inclusion of Floss does not consistently outperform the models without it. This could be attributed to the random factors involved in the training process with Floss, such as the random sampling for periodicity detection and the random shift using the detected periodicity. These factors might prevent the models from consistently leveraging valuable information. Future studies should address this issue to ensure more consistent results.

As depicted in Figure 4, the prediction results of PatchTST and TS2Vec w/o Floss are presented for the *ETTh2* and *weather* datasets. In the long-term forecasting horizon of *ETTh2*, Floss demonstrates its superiority in handling distribution shifts and trend-seasonality features in comparison to TS2Vec. This advantage can be attributed to the enhanced ability of Floss to effectively leverage trend information by regularizing representations in the frequency domain. Figure 4c further demonstrates the superior performance of PatchTST-Floss in both short-term and long-term forecasting tasks, highlighting the significant benefits introduced by Floss in the context of forecasting.

4.2 Unsupervised Time Series Classification with TS2Vec

4.2.1 Experimental Setup. In this section, we combine Floss with the state-of-the-art (SOTA) unsupervised framework TS2Vec [47], which has outperformed several supervised learning frameworks. We utilize the same convolutional encoder as described in [47]. Additionally, we modify the sampling strategy of TS2Vec to create periodic shifts, aligning it with the settings used for the aforementioned multivariate time series forecasting. Following the pretraining phase, we train an SVM classifier with an RBF kernel on top of the instance-level representations to perform predictions.

4.2.2 Public Datasets. We evaluate the effectiveness of our proposed Floss by assessing its classification performance on two widely-used datasets: the UCR archive [6] and UEA archive [3].

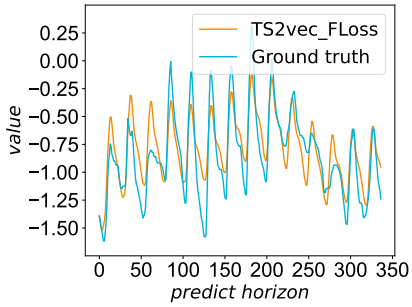
Table 1: Statistics of popular datasets for benchmark.

Datasets	ETTh1&ETTh2	ETTm1 &ETTm2	Electricity	Exchange-Rate	Weather	IL
Variates	7	7	321	8	21	7
Timesteps	17,420	69,680	26,304	7,588	52,696	966
Granularity	1hour	15min	1hour	1day	10min	1week

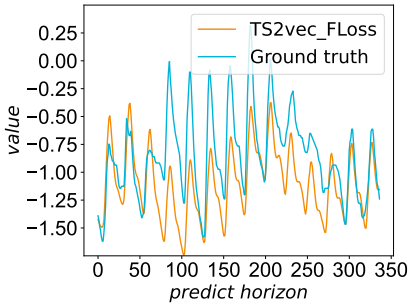
Table 2: Errors of Multivariate Time Series Forecasting. The improved results are in bold.

Dataset		Informer		Informer-Floss		TS2vec		TS2vec-Floss		PatchTST		PatchTST-Floss	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Weather	96	0.427	0.460	**0.277**	**0.370**	1.719	0.921	**1.278**	**0.840**	0.144	0.192	**0.125**	**0.173**
	192	0.346	0.414	0.361	**0.402**	1.650	0.925	**1.360**	**0.882**	0.191	0.241	**0.183**	**0.229**
	336	0.583	0.543	**0.407**	**0.408**	1.949	1.043	**1.318**	**0.876**	0.244	0.280	**0.232**	**0.271**
	720	0.916	0.705	**0.837**	**0.668**	2.718	1.287	**1.559**	**0.972**	0.314	0.331	**0.301**	**0.325**
Exchange	96	0.841	0.746	**0.753**	**0.705**	0.498	0.527	**0.422**	**0.484**	0.099	0.224	0.099	0.225
	192	1.132	0.847	**1.180**	0.859	1.112	0.781	**0.851**	**0.687**	0.210	0.331	0.210	**0.330**
	336	1.475	0.956	1.510	0.974	1.561	0.967	1.571	**0.944**	0.404	0.468	0.424	0.478
	720	2.548	1.328	2.606	1.362	2.688	1.266	**1.860**	**1.052**	1.039	0.769	**0.902**	**0.720**
Electricity	96	0.304	0.393	**0.285**	**0.380**	0.452	0.492	**0.422**	**0.463**	0.135	0.231	**0.129**	**0.228**
	192	0.327	0.417	**0.297**	**0.390**	0.461	0.498	**0.423**	**0.465**	0.150	0.244	**0.149**	**0.242**
	336	0.333	0.422	**0.302**	**0.396**	0.472	0.491	**0.426**	**0.468**	0.165	0.259	**0.159**	0.260
	720	0.351	0.427	**0.325**	**0.406**	0.544	0.547	**0.513**	**0.516**	0.203	0.292	**0.201**	**0.287**
ILI	24	5.940	1.720	**5.460**	**1.580**	3.349	1.168	3.686	1.276	2.883	1.189	2.962	1.200
	36	4.999	1.508	5.300	1.541	3.671	1.244	4.131	1.399	2.986	1.195	**2.850**	**1.169**
	48	5.004	1.542	5.319	1.570	4.150	1.324	4.153	1.364	3.411	1.287	**2.899**	**1.174**
	60	5.403	1.554	5.631	1.589	4.231	1.340	**4.185**	1.359	3.207	1.233	**3.142**	**1.227**
ETTh1	96	0.941	0.769	**0.801**	**0.695**	0.699	0.592	0.804	0.666	0.373	0.402	**0.368**	**0.397**
	192	1.007	0.786	**0.867**	**0.713**	0.789	0.643	0.876	0.704	0.403	0.419	0.403	0.421
	336	1.038	0.784	1.140	0.859	0.907	0.709	0.969	0.750	0.443	0.449	**0.432**	**0.441**
	720	1.144	0.857	1.184	0.883	1.084	0.800	**0.969**	**0.750**	0.482	0.490	**0.451**	**0.472**
ETTh2	96	3.283	1.502	**2.763**	**1.372**	1.034	0.806	1.065	0.808	0.287	0.344	**0.285**	**0.342**
	192	4.371	1.815	**4.110**	**1.713**	1.973	1.118	2.177	1.163	0.363	0.392	**0.359**	**0.386**
	336	4.215	1.642	**3.910**	1.656	2.831	1.319	**2.398**	**1.238**	0.375	0.409	0.376	**0.405**
	720	3.656	1.619	**3.222**	**1.541**	2.561	1.353	2.578	**1.331**	0.411	0.443	**0.399**	**0.428**
ETTm1	96	0.657	0.575	**0.629**	0.582	0.611	0.551	**0.565**	**0.519**	0.282	0.339	**0.281**	**0.328**
	192	0.725	0.619	0.744	0.647	0.675	0.589	**0.616**	**0.553**	0.329	0.369	**0.319**	**0.356**
	336	0.725	0.619	1.053	0.819	0.725	0.621	**0.681**	**0.593**	0.358	0.387	**0.349**	**0.378**
	720	1.133	0.845	**0.997**	**0.778**	0.810	0.671	**0.763**	**0.643**	0.411	0.415	**0.397**	**0.411**
ETTm2	96	0.555	0.462	**0.488**	0.514	0.443	0.495	**0.371**	**0.447**	0.164	0.254	**0.158**	**0.233**
	192	0.695	0.686	0.715	**0.652**	0.615	0.598	**0.546**	**0.561**	0.220	0.294	**0.197**	**0.252**
	336	1.270	0.871	**1.119**	**0.805**	0.975	0.765	**0.863**	**0.721**	0.271	0.327	**0.248**	**0.319**
	720	3.171	1.367	3.414	1.374	2.024	1.093	**1.977**	1.104	0.354	0.381	**0.339**	**0.355**
Avg.		1.868	0.935	1.812	0.912	1.562	0.860	1.449	0.831	0.666	0.465	0.635	0.452
Improvements.				↓ 3.0%	↓ 2.4%			↓ 7.2%	↓ 3.4%			↓ 4.6%	↓ 2.8%

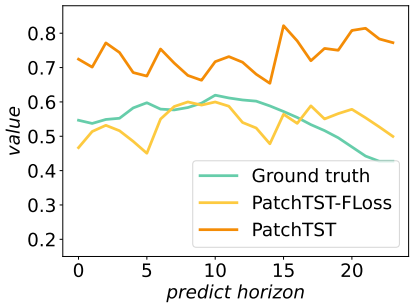
The UCR archive consists of 128 univariate datasets, while the UEA archive contains 30 multivariate datasets. For each dataset considered, we utilize its original train/test split. We conduct unsupervised training of an encoder using the train set of each dataset. Subsequently, we train an SVM classifier with a RBF kernel on top of the learned features, utilizing the train labels of the dataset. Finally, we output the corresponding classification score on the test set. For the hyperparameter settings, batch size is 16, the contrastive loss weight is 1, Floss weight is 1.



(a) TS2Vec-Floss forecasting results on *ETTh2*



(b) TS2Vec forecasting results on *ETTh2*



(c) PatchTST and PatchTST-Floss forecasting results on *weather*

Figure 4: Illustration of the long-term forecasting output of model w/o Floss on *ETTm2* and *weather* datasets (Y-axis: forecasting horizon).

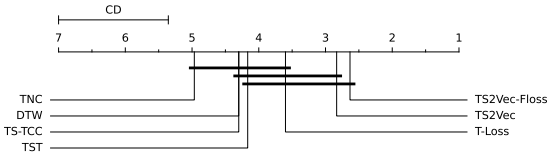


Figure 5: Critical Difference (CD) diagram of representation learning methods on time series classification tasks with a confidence level of 95%.

4.2.3 Compared Baselines. We perform comprehensive experiments on time series classification to assess the classification performance of our approach, in comparison to other unsupervised time series representation models, namely T-Loss [10], TS-TCC [9], TST [49], and TNC [32]. Additionally, we include DTW (Dynamic Time Warping) [23] as a baseline, employing a one-nearest-neighbor classifier with DTW as the distance measure.

Table 3: Time series classification results compared to other time series representation methods on 125 UCR datasets and 29 UEA datasets.

Method	125 UCR datasets	29 UEA datasets
DTW	0.727	0.650
TNC	0.761	0.677
TST	0.641	0.635
TS-TCC	0.757	0.682
T-Loss	0.806	0.675
TS2Vec	0.830	0.712
TS2Vec-Floss	**0.849**	**0.739**

4.2.4 Experimental Results. The evaluation results are summarized in Table 3. Floss demonstrates a significant improvement compared to other representation learning methods on both the UCR and UEA datasets. Specifically, Floss achieves an average increase of 2.3% in classification accuracy over TS2Vec across 125 UCR datasets and 3.0% across 29 UEA datasets. It is important to note that the periodicity detection module is applicable to all UCR and UEA datasets, and comprehensive results of TS2Vec-Floss on all datasets can be found in the supplementary materials. Critical Difference diagram [8] for Nemenyi tests on all datasets (including 125 UCR and 29 UEA datasets) is presented in Figure 5, where classifiers that are not connected by a bold line are significantly different in average rank. Unlike existing baselines that neglect periodic information, Floss utilizes hierarchical frequency domain comparison between different periodic views, resulting in enhanced performance.

4.3 Unsupervised Time Series Classification with TS-TCC

4.3.1 Experimental Setup. We combine Floss with another representative model for time series representation called TS-TCC [9]. To evaluate our model, we conduct human activity recognition, sleep stage classification, and epileptic seizure prediction tasks using open-source datasets. Following the approach of TS-TCC, we perform pre-training and downstream task fine-tuning for 40 epochs. During the pre-training phase, we incorporate Floss with the contrastive loss function of TS-TCC. In contrast to the TS2Vec setup, we introduce a separate periodic augmentation alongside the jitter and scale augmentation of TS-TCC. Moreover, Floss is computed based solely on the original and periodic views of the time series data. The encoder is trained using Adam with a weighted sum of Floss and the original loss of TS-TCC. We maintain the same hyperparameters as those reported in [9]. For the loss weights, the original loss weight is 0.3 and Floss weight is 2.

4.3.2 Public Datasets. We assess the classification performance of our proposed Floss by evaluating it on three widely-used datasets: 1. UCI HAR dataset [2]: This dataset contains sensor readings for 30 subjects performing 6 activities. The sample rate of the HAR dataset is 60Hz. 2.Sleep-EDF [13]: This dataset includes whole-night PSG sleep recordings, with a sampling rate of 100Hz. 3. The Epileptic Seizure Recognition dataset [1]: This dataset consists of EEG recordings from 500 subjects, where the brain activity was recorded for each subject for 23.6 seconds. We split the data into 60%, 20%, and 20% for training, validation, and testing, respectively. For

the Sleep-EDF dataset, we perform a subject-wise split to prevent overfitting. We repeat the experiments five times using five different seeds. During the fine-tuning phase, we train a linear classifier (a single MLP layer) on top of a frozen self-supervised pretrained encoder model to perform classification.

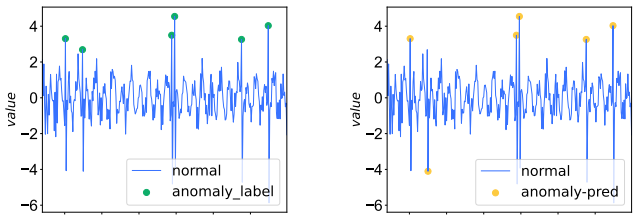
4.3.3 Experimental Results. We report the accuracy (ACC) and macro F1 score (MF1) of the TS-TCC-Floss, raw TS-TCC [9], CPC [25] and SimCLR [5] in Table 4. Similar to the findings observed in the TS2Vec experiments, the integration of Floss yields significant enhancements in the performance of TS-TCC. Notably, an intriguing aspect emerges when examining the three datasets employed in this study, wherein the sampling periods are comparatively short. Intuitively, discerning the presence of short-term periodic information in these datasets poses a formidable challenge. However, employing Floss still yields notable improvements across these datasets. This phenomenon can be attributed to the inherent capacity of Floss to autonomously detect periodicity, thereby effectively capturing imperceptible quasi-periodic variations within the data. Consequently, Floss exhibits an automatic mechanism for augmenting the representational quality of existing models, thereby advancing their efficacy.

4.4 Unsupervised Anomaly Detection

4.4.1 Experimental Setup. For anomaly detection, we follow the streaming evaluation protocol, where the task is to determine whether the last point t is an anomaly. As same as in [47], we define the anomaly score as the dissimilarity between the representations computed from the original series and the one with a mask at the last time point. We use the same computation strategy as described in [47] to compute anomalies. Two public datasets are used to evaluate our model. *Yahoo*[1] is a benchmark dataset for anomaly detection, which includes 367 hourly sampled time series with tagged anomaly points. *KPI* [27] includes multiple minutely sampled real KPI curves from various Internet companies. In the normal setting, each time series sample is split into two halves according to the time order, where the first half is used for unsupervised training and the second half is used for evaluation. We also evaluate the cold-start problem, in which the TS2Vec and Floss encoder are trained on the *ItalyPowerDemand* dataset from the UCR, as *ItalyPowerDemand* exhibits daily periodicity. We use precision, recall and F1-score to measure the performance of anomaly detection. For normal settings, batch size is set to 16, Floss weight is 1, contrastive loss weight is 0.6. For Yahoo(*Cold-start*)and KPI(*Cold-start*), Batch size is 16, Floss weight is 1, contrastive loss weight is 1.

4.4.2 Experimental Results. The anomaly detection performance of TS2Vec-Floss, TS2Vec, and a strong unsupervised learning baseline SR [27] are presented in Table 5. In the normal setting, Floss improves the F1 score by 1.19% on the *Yahoo* dataset and 1.08% on the *KPI* dataset compared to TS2Vec. This indicates that Floss is more sensitive to outliers in time series, as it captures periodic dynamics and expresses fine-grained information through hierarchical pooling. In the cold start setting, the improvement of Floss

[1]https://yahooresearch.tumblr.com/post/114590420346/a-benchmark-dataset-for-time-series-anomaly



(a) Anomaly detection labels of Yahoo

(b) Anomaly predicted by Floss of Yahoo

Figure 6: Anomaly detection results

on both datasets is even more noticeable (about 10% on F1 score), demonstrating its ability to capture general periodic invariance with strong transferability.

We also provide visualizations of the anomaly detection performed by Floss in Figure 6. In both examples, we observe that Floss accurately identifies all anomalies. It is worth noting that the only negative result obtained by Floss is still in close proximity to the corresponding ground truth anomaly time point.

4.5 Unsupervised Anomaly Detection with other Models

4.5.1 Experimental Setup. In this study, we perform experiments on two extensively utilized anomaly detection datasets: MSL (Mars Science Laboratory rover) [16] and SMD (Server Machine Dataset) [30]. We selected these two datasets due to their pronounced periodic patterns. Adhering to the pre-processing techniques outlined in Anomaly Transformer [45], we divided the dataset into sequential, non-overlapping segments using a sliding window approach. Subsequently, we employed a deep learning model to reconstruct the input samples, with the resulting reconstruction error serving as the inherent anomaly indicator. To ensure equitable comparisons, we solely modified the base models for reconstruction, employing the conventional reconstruction error as the standardized anomaly criterion across all experiments. Each dataset consists of training and testing subsets, with validation subsets identical to the testing subsets. Anomalies are only labeled within the testing subset. The calculation of Floss is integrated into the 'anomaly_detection' method of each model. Initially, we perform periodicity detection on the input data and extract periodic segments. Subsequently, we extract features from these segments and calculate the Floss. Finally, the Floss is incorporated into the model training process. Throughout these experiments, the Floss weight is set to 1, and the reconstruction loss weight is set to 0.3.

After training the model, we analyze the training data within a gradient-free context. For each data batch, we employ the trained model to reconstruct it and calculate the reconstruction error scores. To establish the anomaly threshold, we aggregate scores from both the training and test datasets. This combined score assists in determining the threshold, based on a predefined anomaly ratio. Subsequently, we compare the test data scores with the threshold to identify anomalies. Scores exceeding the threshold are classified as anomalies, while those falling below it are categorized as normal.

Table 4: Time series classification results compared to other time series representation methods on HAR, Sleep-EDF and Epilepsy.

Datasets	HAR		Sleep-EDF		Epilepsy	
Metric	ACC	MF1	ACC	MF1	ACC	MF1
CPC	83.85 ± 1.51	83.27 ± 1.66	82.82 ± 1.68	**73.94 ± 1.75**	96.61 ± 0.43	94.44 ± 0.69
SimCLR	80.97 ± 2.46	80.19 ± 2.64	78.91 ± 3.11	68.60 ± 2.71	96.05 ± 0.34	93.53 ± 0.63
TS-TCC	90.37 ± 0.34	90.38 ± 0.39	83.00 ± 0.71	73.57 ± 0.74	97.23 ± 0.10	95.54 ± 0.08
TS-TCC-Floss	**90.86 ± 0.34**	**90.56 ± 0.35**	**83.70 ± 0.45**	73.53 ± 0.39	**97.41 ± 0.17**	**97.75 ± 0.00**

Table 5: Univariate time series anomaly detection results.

Dataset	Yahoo			KPI		
Metric	F1	Prec.	Rec.	F1	Prec.	Rec.
SR	0.563	0.451	0.747	0.622	0.647	**0.598**
TS2Vec	0.745	0.729	0.762	0.677	0.929	0.533
TS2Vec-FLoss	**0.754**	**0.752**	**0.763**	**0.799**	**0.946**	0.559
Cold-start:						
SR	0.529	0.404	0.765	0.666	0.637	**0.697**
TS2Vec	0.726	0.692	0.763	0.676	0.907	0.540
TS2Vec-FLoss	**0.734**	**0.706**	**0.769**	**0.741**	**0.942**	0.594

4.5.2 Models Improved by Floss. We consider four notable multivariate time series forecasting models as featured in [41]: 1).FEDformer [53]: This model combines a Transformer architecture with the seasonal-trend decomposition method. 2).TimesNet [41]: It employs Fast Fourier Transform (FFT) to convert the time series into a 2D representation, utilizing CNNs as the foundational framework. 3). Reformer [18]: This variant of the Transformer replaces the conventional dot-product attention mechanism with a locality-sensitive hashing approach. 4). A conventional Transformer [36].

4.5.3 Results. Table 6 illustrates that Floss continues to enhance anomaly detection performance, yielding improvements for the selected model in most instances. We have summarized some intriguing observations as follows: 1). Floss demonstrates a notable improvement in the F1 scores for nearly all models, with the exception being TimesNet on the SMD dataset. This discrepancy could potentially arise from TimesNet's adept utilization of periodic information through its inherent FFT block. 2). Remarkably, the Transformer-Floss combination attains the highest F1 score on the MSL dataset, surpassing even the more intricate TimesNet model. This outcome suggests that Floss can imbue a simpler model with robust time series processing capabilities, offering valuable insights for designing models in the context of anomaly detection tasks.

We demonstrate the reconstruction effects after incorporating Floss into the Transformer in Figure 7. It can be observed that when anomalies occur, the Transformer model with Floss exhibits larger reconstruction errors compared to the regular Transformer model. Floss, to some extent, preserves the consistency of periodic observations in the spectrum. Many anomalies often manifest as significant changes in certain parts of the spectrum. Therefore, Floss, by maintaining the consistency of periodic spectral patterns, is advantageous for anomaly detection.

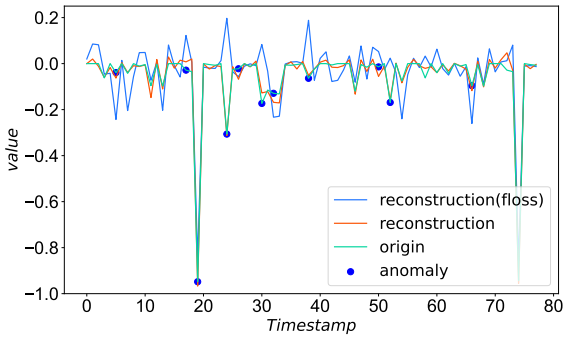


Figure 7: Visualization of the reconstruction, true value and anomaly.

4.6 Detailed Study of Floss

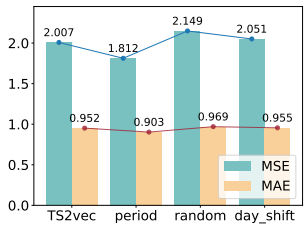
As Floss is designed as a plug-in loss function, there can be various instances with different implementation choices for each module. In this section, a comprehensive analysis and comparison of different instances of Floss are conducted. In the following discussions, we consider the simplest TS2Vec as the baseline and compare it with other variants on multivariate time series forecasting for Weather, Exchange, ILI and Ettm1. Furthermore, we employ a fixed set of hyperparameters to ensure a fair comparison. It is worth noting that some results may appear worse than those reported in Table 2 because we only presented the best results in Table 2

4.6.1 Effects of periodic detection module. We initiated our investigation by examining the influence of the period detection module on the model. A comparative analysis was conducted between Floss and two alternative models, namely random and day shift. In Figure 8a, 'random' signifies the utilization of random augmentation during each comparison with Floss, while 'day shift' denotes the shifting of time series by one day at each step, operating under the implicit assumption that all time series exhibit a daily periodicity. The outcomes unveiled that both models incorporating random shifting and day shifting exhibited inferior performance compared to the TS2Vec model.

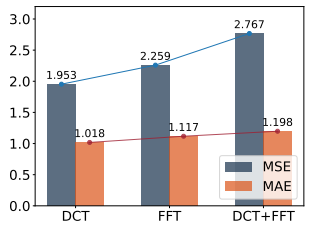
Since Floss assumes that the representation of periodic shifts is similar in the frequency domain, augmenting time series with

Table 6: Anomaly detection task. We calculate the accuracy, precision, recall and F1 scores for each dataset.

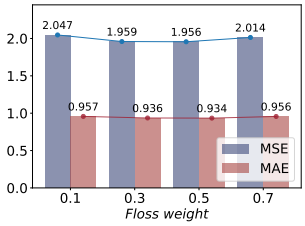
Dataset	MSL				SMD			
Metric	Acc.	Prec.	Rec.	F1	Acc.	Prec.	Rec.	F1
FEDformer	0.9673	0.7714	0.7679	0.7857	0.9763	0.7732	0.6094	0.6816
FEDformer-Floss	0.9651	**0.9059**	0.7465	**0.8185**	**0.9781**	**0.7846**	**0.6508**	**0.7114**
TimesNet	0.9647	0.8955	0.7529	0.8180	0.9877	0.8788	0.8154	0.8459
TimesNet-Floss	**0.9648**	**0.8959**	**0.7541**	**0.8187**	0.9867	0.8684	0.8008	0.8332
Reformer	0.9638	0.9014	0.7372	0.8111	0.9780	0.7832	0.6524	0.7118
Reformer-Floss	**0.9647**	**0.9055**	**0.7430**	**0.8163**	**0.9781**	0.7832	**0.6538**	**0.7127**
Transformer	0.9634	0.8977	0.7366	0.8093	0.9780	0.7832	0.6524	0.7118
Transformer-Floss	**0.9652**	**0.9062**	**0.7470**	**0.8189**	**0.9781**	0.7828	**0.6537**	**0.7125**



(a) Effect of the period detection module on forecasting errors .



(b) FFT and DCT: peridoicity is detected by FFT, DCT+FFT: periodicity is dtected by DCT.



(c) Effect of Floss weights on forecasting errors.

Figure 8: Ablation results.

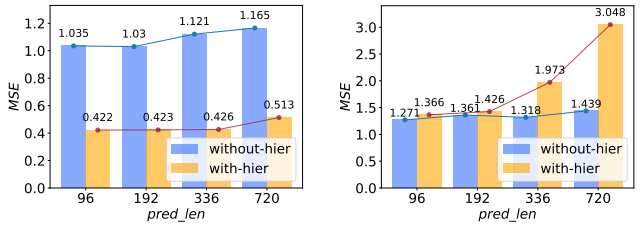
random shifts or assuming a daily shift might not effectively capture the underlying patterns and periodic behavior. These findings suggest that considering generic shifts or assuming a specific daily pattern might overlook the nuanced dynamics of the time series. Notably, it was observed that only the model incorporating the period detection module for augmentation outperformed TS2Vec. This highlights the critical role played by the period detection module in enhancing the model's performance.

4.6.2 Combination of DCT and FFT. We also conducted an examination of the combination of Fast Fourier Transform (FFT) and Discrete Cosine Transform (DCT) in relation to period identification and Floss computation. In Figure 8b, 'FFT' and 'DCT' signify the utilization of FFT and DCT for Floss calculation, respectively. Notably, both approaches employ FFT for periodic detection. On the other hand, 'DCT+FFT' indicates the application of DCT for period identification and FFT for Floss computation. Our investigation yielded noteworthy results, unveiling the significance of the combination. It was observed that employing FFT for period identification, while leveraging DCT for spectral density computation, yielded the most optimal outcomes in terms of performance.

4.6.3 Effects of Floss weights. Floss operates in conjunction with the loss of other models. Our encoder is trained using a weighted sum of Floss and other loss functions. Assigning a higher weight to Floss indicates a greater reliance on capturing periodic invariances. To investigate the impact of the Floss weight, we set the contrastive loss weight of TS2Vec to 0.5 and evaluate the model's performance with different loss weights on three datasets. The results are presented in Figure 8c. The findings demonstrate the robustness of our proposed method to the choice of weight. The performance of the model remains consistent across various weight settings. However, upon closer analysis, we identify that the weight range between 0.3 and 0.5 yields the best performance.

4.6.4 Effects of hierarchical Floss computation. As described in Section 3.2, we employ a hierarchical Floss computation strategy to allocate greater weights to the low-frequency components. However, it is noteworthy that employing hierarchical Floss computation may not be necessary for all datasets. The performance comparison without hierarchical computation is presented in Figure 9. Specifically, our experimentation on the electricity dataset demonstrates a substantial enhancement in model performance when utilizing hierarchical Floss computation. In contrast, for the weather dataset, we observed that refraining from hierarchical Floss computation actually yielded superior outcomes. When employing hierarchical computation, we tend to focus more on capturing the similarities in the low-frequency components. On the other hand, without employing hierarchical computation, we treat all frequency components equally, including the high-frequency components. This observation suggests that in datasets such as weather, after undergoing periodic variations, the abstract representation of the high-frequency components remains relatively unchanged. Preserving all frequencies becomes more effective for such data.

Moreover, we observed a significant improvement in long-term forecasting performance when hierarchical Floss computation was not employed for weather dataset. This finding suggests that for the weather dataset, the long-term variation trend may be concealed



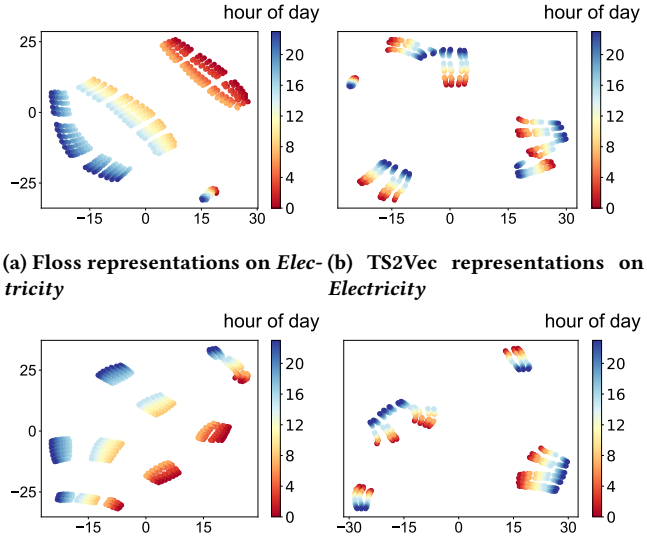
(a) Effect of the hierarchical Floss computation on Electricity dataset.

(b) Effect of the hierarchical Floss computation on Weather dataset .

Figure 9: Effect of hierarchical Floss computation.

within the unchanged high-frequency components under periodic shifts. These phenomena call for further in-depth research to design more robust models capable of capturing these patterns.

4.6.5 Representation visualization. Figure 10 displays the t-SNE embedding of TS2Vec-Floss and Floss on nine consecutive days of the *Electricity* and *ETTh1* datasets. These datasets are known to exhibit pronounced daily periodicity. Consequently, the automatic periodic detection module is anticipated to capture this strong periodic pattern. In this visualization, the model with Floss produces a more periodic cloud structure, characterized by a reduced presence of easily distinguishable hour-of-day groupings.



(a) Floss representations on *Electricity*

(b) TS2Vec representations on *Electricity*

(c) Floss representations on *ETTh1*

(d) TS2Vec representations on *ETTh1*

Figure 10: T-SNE visualizations of the learned representations of TS2Vec-Floss and TS2Vec on *Electricity* and *ETTh1*. Different colors represent different hours of day.

4.6.6 Accuracy of Periodicity Detection. We provide a case study (informer-Floss for Electricity) of the periodicity detection module in Figure 11. We can observe that Floss can accurately capture

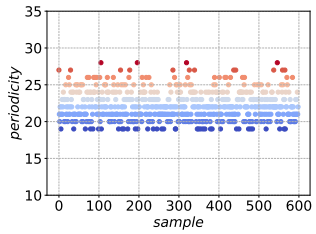


Figure 11: Periodicity Detection Results with informer-Floss for Electricity.

the periodicities. Moreover, most of the detected periodicities are approximately equal to 24 hours (1 day), which supports our motivation in adopting automatic periodicity detection for representation learning.

5 CONCLUSION

In this study, we addressed the challenge of effectively capturing periodic or quasi-periodic dynamics present in real-world time series data using deep learning approaches. While deep learning has shown impressive performance in various application domains, it often struggles to adequately represent the underlying periodic behaviors in time series data. To bridge this gap, we introduced an unsupervised method called Floss. Floss is designed to automatically detect major periodicities in time series data and utilizes periodic shift and spectral density similarity measures to learn meaningful representations with periodic consistency in the frequency domain. By seamlessly incorporating Floss into supervised, semi-supervised, and unsupervised learning frameworks, we demonstrated its versatility and ability to enhance time series analysis tasks.

Our extensive experiments on common time series analysis tasks showcased the effectiveness of Floss. It outperformed state-of-the-art deep learning models, validating its capability to automatically discover periodic dynamics. The results underscore the importance of considering domain-specific knowledge about periodic behaviors to enrich the learned representations in deep learning models.

For future work, exploring advanced modeling techniques that can effectively capture the hidden long-term patterns in complex data such as weather data remains a promising direction. In the weather dataset, we observed a significant improvement in long-term forecasting performance when hierarchical Floss computation was not employed. his finding suggests that for some datasets, the long-term variation trend may be concealed within the unchanged high-frequency components under periodic shifts. For future work, exploring advanced modeling techniques that can effectively capture the hidden long-term patterns in weather data remains a promising direction. This may involve the incorporation of domain-specific knowledge, such as external factors, to enhance the modeling process. Furthermore, extending the research to consider more complex and dynamic scenarios, such as time series prediction under extreme weather events, could present new challenges and opportunities for advancing time series analysis. Floss solely addresses the frequency domain similarity of the model concerning temporal periodicity. Integrating state-of-the-art techniques, such as Graph Neural Networks (GNNs) and graph spectral analysis,

holds promise for modeling inter time series invariance and optimizing time series analysis performance. By leveraging GNNs and graph spectral analysis, we can gain a deeper understanding of the relationships between multiple time series, capturing intricate temporal dependencies and interdependencies among time series.

ACKNOWLEDGMENTS

This work was supported by National Natural Science Foundation of Sichuan Province (Grant No.2023NSFSC1423), the Tianfu Emei Plan of Sichuan Province, and the Fundamental Research Funds for the Central Universities.

REFERENCES

[1] Ralph G Andrzejak, Klaus Lehnertz, Florian Mormann, Christoph Rieke, Peter David, and Christian E Elger. 2001. 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 (2001), 061907.

[2] Davide Anguita, Alessandro Ghio, Luca Oneto, Xavier Parra, Jorge Luis Reyes-Ortiz, et al. 2013. A public domain dataset for human activity recognition using smartphones.. In *Esann*, Vol. 3. 3.

[3] Anthony Bagnall, Hoang Anh Dau, Jason Lines, Michael Flynn, James Large, Aaron Bostrom, Paul Southam, and Eamonn Keogh. 2018. The UEA multivariate time series classification archive, 2018. *arXiv preprint arXiv:1811.00075* (2018).

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

[5] Xinlei Chen and Kaiming He. 2021. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 15750–15758.

[6] Hoang Anh Dau, Anthony Bagnall, Kaveh Kamgar, Chin-Chia Michael Yeh, Yan Zhu, Shaghayegh Gharghabi, Chotirat Ann Ratanamahatana, and Eamonn Keogh. 2019. The UCR time series archive. *IEEE/CAA Journal of Automatica Sinica* 6, 6 (2019), 1293–1305.

[7] Jan G De Gooijer and Rob J Hyndman. 2006. 25 years of time series forecasting. *International journal of forecasting* 22, 3 (2006), 443–473.

[8] Janez Demšar. 2006. Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine learning research* 7 (2006), 1–30.

[9] Emadeldeen Eldele, Mohamed Ragab, Zhenghua Chen, Min Wu, Chee Keong Kwoh, Xiaoli Li, and Cuntai Guan. 2021. Time-Series Representation Learning via Temporal and Contextual Contrasting. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*. 2352–2359.

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

[11] Wayne A Fuller. 2009. *Introduction to statistical time series*. John Wiley & Sons.

[12] Shaghayegh Gharghabi, Chin-Chia Michael Yeh, Yifei Ding, Wei Ding, Paul Hibbing, Samuel LaMunion, Andrew Kaplan, Scott E Crouter, and Eamonn Keogh. 2019. Domain agnostic online semantic segmentation for multi-dimensional time series. *Data mining and knowledge discovery* 33 (2019), 96–130.

[13] 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. 2000. PhysioBank, PhysioToolkit, and PhysioNet: components of a new research resource for complex physiologic signals. *circulation* 101, 23 (2000), e215–e220.

[14] Michael Gutmann and Aapo Hyvärinen. 2010. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 297–304.

[15] Tao Hong, Pierre Pinson, Shu Fan, Hamidreza Zareipour, Alberto Troccoli, and Rob J Hyndman. 2016. Probabilistic energy forecasting: Global energy forecasting competition 2014 and beyond. , 896–913 pages.

[16] Kyle Hundman, Valentino Constantinou, Christopher Laporte, Ian Colwell, and Tom Soderstrom. 2018. Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*. 387–395.

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

[18] Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. 2019. Reformer: The Efficient Transformer. In *International Conference on Learning Representations*.

[19] Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. 2018. Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting. In *International Conference on Learning Representations*. https://openreview.net/forum?id=SJiHXGWAZ

[20] Zachary C Lipton, David C Kale, Randall Wetzel, et al. 2016. Modeling missing data in clinical time series with rnns. *Machine Learning for Healthcare* 56 (2016), 253–270.

[21] Laura A McSweeney. 2006. Comparison of periodogram tests. *Journal of Statistical Computation and Simulation* 76, 4 (2006), 357–369.

[22] LIU Minhao, Ailing Zeng, LAI Qiuxia, Ruiyuan Gao, Min Li, Jing Qin, and Qiang Xu. 2021. T-WaveNet: A Tree-Structured Wavelet Neural Network for Time Series Signal Analysis. In *International Conference on Learning Representations*.

[23] Meinard Müller. 2007. Dynamic time warping. *Information retrieval for music and motion* (2007), 69–84.

[24] Yuqi Nie, Nam H Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. 2023. A Time Series is Worth 64 Words: Long-term Forecasting with Transformers. In *The Eleventh International Conference on Learning Representations*. https://openreview.net/forum?id=Jbdc0vTOcol

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

[26] John Paparrizos, Yuhao Kang, Paul Boniol, Ruey S Tsay, Themis Palpanas, and Michael J Franklin. 2022. TSB-UAD: an end-to-end benchmark suite for univariate time-series anomaly detection. *Proceedings of the VLDB Endowment* 15, 8 (2022), 1697–1711.

[27] Hansheng Ren, Bixiong Xu, Yujing Wang, Chao Yi, Congrui Huang, Xiaoyu Kou, Tony Xing, Mao Yang, Jie Tong, and Qi Zhang. 2019. Time-series anomaly detection service at microsoft. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*. 3009–3017.

[28] David Salinas, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. 2020. DeepAR: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting* 36, 3 (2020), 1181–1191.

[29] Huan Song, Deepta Rajan, Jayaraman Thiagarajan, and Andreas Spanias. 2018. Attend and diagnose: Clinical time series analysis using attention models. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 32.

[30] Ya Su, Youjian Zhao, Chenhao Niu, Rong Liu, Wei Sun, and Dan Pei. 2019. Robust anomaly detection for multivariate time series through stochastic recurrent neural network. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*. 2828–2837.

[31] Andrzej Tarczynski and Najib Allay. 2004. Spectral analysis of randomly sampled signals: suppression of aliasing and sampler jitter. *IEEE Transactions on Signal Processing* 52, 12 (2004), 3324–3334.

[32] Sana Tonekaboni, Danny Eytan, and Anna Goldenberg. 2020. Unsupervised Representation Learning for Time Series with Temporal Neighborhood Coding. In *International Conference on Learning Representations*.

[33] Machiko Toyoda, Yasushi Sakurai, and Yoshiharu Ishikawa. 2013. Pattern discovery in data streams under the time warping distance. *The VLDB Journal* 22 (2013), 295–318.

[34] Ruey S Tsay. 2005. *Analysis of financial time series*. John wiley & sons.

[35] Jacob T VanderPlas and Željko Ivezic. 2015. Periodograms for multiband astronomical time series. *The Astrophysical Journal* 812, 1 (2015), 18.

[36] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[37] Michail Vlachos, Philip Yu, and Vittorio Castelli. 2005. On periodicity detection and structural periodic similarity. In *Proceedings of the 2005 SIAM international conference on data mining*. SIAM, 449–460.

[38] Qingsong Wen, Kai He, Liang Sun, Yingying Zhang, Min Ke, and Huan Xu. 2021. RobustPeriod: Robust time-frequency mining for multiple periodicity detection. In *Proceedings of the 2021 International Conference on Management of Data*. 2328–2337.

[39] Gerald Woo, Chenghao Liu, Doyen Sahoo, Akshat Kumar, and Steven Hoi. 2021. CoST: Contrastive Learning of Disentangled Seasonal-Trend Representations for Time Series Forecasting. In *International Conference on Learning Representations*.

[40] Gerald Woo, Chenghao Liu, Doyen Sahoo, Akshat Kumar, and Steven Hoi. 2022. ETSformer: Exponential Smoothing Transformers for Time-series Forecasting. *arXiv preprint arXiv:2202.01381* (2022).

[41] Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. 2023. TimesNet: Temporal 2D-Variation Modeling for General Time Series Analysis. In *The Eleventh International Conference on Learning Representations*. https://openreview.net/forum?id=ju_Uqw384Oq

[42] Xinle Wu, Dalin Zhang, Chenjuan Guo, Chaoyang He, Bin Yang, and Christian S Jensen. 2021. AutoCTS: Automated correlated time series forecasting. *Proceedings of the VLDB Endowment* 15, 4 (2021), 971–983.

[43] Yuankai Wu, Huachun Tan, Lingqiao Qin, Bin Ran, and Zhuxi Jiang. 2018. A hybrid deep learning based traffic flow prediction method and its understanding. *Transportation Research Part C: Emerging Technologies* 90 (2018), 166–180.

[44] Z Wu, S Pan, G Long, J Jiang, and C Zhang. 2019. Graph WaveNet for Deep Spatial-Temporal Graph Modeling. In *The 28th International Joint Conference on Artificial Intelligence (IJCAI)*. International Joint Conferences on Artificial Intelligence Organization.

[45] Jiehui Xu, Haixu Wu, Jianmin Wang, and Mingsheng Long. 2021. Anomaly Transformer: Time Series Anomaly Detection with Association Discrepancy. In *International Conference on Learning Representations*.

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

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

[48] Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. 2023. Are Transformers Effective for Time Series Forecasting? *Proceedings of the AAAI Conference on Artificial Intelligence*.

[49] George Zerveas, Srideepika Jayaraman, Dhaval Patel, Anuradha Bhamidipaty, and Carsten Eickhoff. 2021. A Transformer-Based Framework for Multivariate Time Series Representation Learning *(KDD '21)*. Association for Computing Machinery, New York, NY, USA, 2114–2124. https://doi.org/10.1145/3447548.3467401

[50] Xiyuan Zhang, Xiaoyong Jin, Karthick Gopalswamy, Gaurav Gupta, Youngsuk Park, Xingjian Shi, Hao Wang, Danielle C Maddix, and Yuyang Wang. 2022. First De-Trend then Attend: Rethinking Attention for Time-Series Forecasting. *arXiv preprint arXiv:2212.08151* (2022).

[51] Xiang Zhang, Ziyuan Zhao, Theodoros Tsiligkaridis, and Marinka Zitnik. 2022. Self-supervised contrastive pre-training for time series via time-frequency consistency. *arXiv preprint arXiv:2206.08496* (2022).

[52] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. 2021. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 11106–11115.

[53] Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. 2022. FEDformer: Frequency enhanced decomposed transformer for long-term series forecasting. *arXiv preprint arXiv:2201.12740* (2022).