*applied sciences*

MDPI

*Article*

# Modeling Trend and Seasonality in Contrastive Learning for Time-Series Forecasting

**Cheng-Ru Chou [1,†], Yen-Ching Lu [1,†], Pei-Xuan Li [1] and Hsun-Ping Hsieh [1,2,*]**

[1] Department of Electrical Engineering, National Cheng Kung University, Tainan 70101, Taiwan; n26140870@gs.ncku.edu.tw (C.-R.C.); n26111986@gs.ncku.edu.tw (Y.-C.L.); n28121107@gs.ncku.edu.tw (P.-X.L.)
[2] Academy of Innovative Semiconductor and Sustainable Manufacturing, National Cheng Kung University, Tainan 70101, Taiwan
* Correspondence: hphsieh@mail.ncku.edu.tw
† These authors contributed equally to this work.

**Abstract**

Self-supervised contrastive learning has recently shown promise for time-series representation learning, yet most existing methods treat sequences holistically and leave trend and seasonal components entangled, limiting their effectiveness for long-horizon multivariate forecasting. We study decomposition-aware representation learning for time-series forecasting without negative pairs. We propose the **T**rend-**S**eason **C**ontrastive **L**earner (TSCL), a Siamese framework that decomposes each series into trend, seasonality, and residual components, encodes trend and seasonality with dedicated encoders and a learnable Fourier layer, and optimizes a positive-pair contrastive objective over component-wise representations. Experiments on five public benchmarks (ETTh1, ETTh2, ETTm1, ETTm2, and Weather) show that TSCL consistently improves downstream forecasting across prediction horizons. Averaged over all datasets and horizons, TSCL achieves **0.489** MSE and **0.488** MAE, yielding an about **20–30%** lower error than representative contrastive baselines (e.g., SimTS and CoST). Paired *t*-tests further confirm that the improvements are statistically significant in most settings. These results indicate that decomposition-aware contrastive learning yields robust and generalizable representations for long-horizon forecasting across diverse temporal resolutions.

**Keywords:** time-series prediction; contrastive learning; feature extraction

Check for updates

Academic Editors: Pierfrancesco Bellini, Haiping Xu and Enrico Collini

Received: 24 December 2025
Revised: 31 January 2026
Accepted: 27 February 2026
Published: 5 March 2026

**Copyright:** © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

## 1. Introduction

Time-series data is widely used in areas such as finance, energy management, medical diagnostics [1], and traffic analysis. However, it is difficult to identify and annotate due to its complexity and diversity. To overcome these limitations and fully leverage unlabeled data, self-supervised learning has emerged as a promising approach for representation learning, excelling in fields such as computer vision, natural language processing, and time-series analysis [2]. Although contrastive learning has proven effective, it remains underexplored in time-series analysis due to the difficulty of extracting meaningful features from complex patterns [3]. Recent contrastive learning approaches for time-series prediction have shown promise [3], but often suffer from overfitting and noise amplification due to excessive model complexity. In addition, the entanglement of trend and seasonal components can hinder effective feature learning.

Recent methods such as CoST [4] and SimTS [5] explore contrastive pretraining for time-series data. CoST disentangles trend and seasonal components using handcrafted temporal and frequency transformations, while SimTS removes negative-pair construction through a Siamese objective. However, existing methods largely treat the series holistically without explicit decomposition, leaving trend and seasonal structure entangled during representation learning. This may limit their suitability for downstream forecasting tasks, where periodic and trend-driven behaviors are important. Inspired by CoST and SimTS, we propose the **T**rend-**S**eason **C**ontrastive **L**earner (TSCL), which decomposes time series into trend, seasonality, and residual components before feature extraction. This decomposition enhances the ability of the feature extractor to effectively capture meaningful features. For seasonal data, a learnable Fourier layer improves representation quality. TSCL also employs a contrastive loss based on Siamese Neural Networks (SNNs) [6,7], focusing on learning the similarity of positive pairs without strictly defining negative pairs. Experiments on five real-world datasets confirm that this method enhances contrastive learning in time-series prediction.

The contributions of our work are as follows:

- We propose the **T**rend-**S**eason **C**ontrastive **L**earner (TSCL), which leverages decomposed seasonality and trend representations with a Siamese architecture to enhance effective learning of trend and seasonality features through contrastive learning without requiring negative pairs.
- The proposed method achieves state-of-the-art performance on five real-world benchmark datasets, reducing MSE and MAE in multivariate time-series forecasting tasks.

## 2. Related Work

### *2.1. Contrastive Learning for Time-Series Forecasting*

Contrastive learning has achieved promising results in representation learning for unlabeled time-series data. Early methods such as T-loss [8] and TNC [9] construct positive and negative pairs based on temporal proximity and subsequence sampling. TS-TCC [3] introduces augmented views for mutual prediction, while TS2Vec [2] applies hierarchical multi-scale contrastive objectives to learn context-dependent representations. Recent studies extend contrastive learning into new domains: CoST [4] performs disentangled contrastive learning in both the time and frequency domains, InfoTS [10] incorporates meta-learning to refine augmentation, and SimTS [5] removes negative-pair construction via a Siamese contrastive objective.

Despite these advancements, most contrastive approaches emphasize representation quality rather than downstream forecasting performance, and largely treat the time series holistically without explicit decomposition. This may leave slowly varying trend components and periodic seasonal patterns entangled in the learned representation space, limiting their effectiveness for multivariate and long-horizon forecasting tasks.

### *2.2. Decomposition and Hybrid Forecasting Approaches*

Decomposition techniques aim to enhance interpretability and forecasting performance by separating trend, seasonal, and residual components [11]. Classical methods include additive and multiplicative decomposition [12] and wavelet transforms [13]. Matrix factorization approaches such as SVD and PCA [14] extract low-dimensional structure to support predictive models.

More recent works integrate decomposition with forecasting models for specific domains. Yuan et al. [15] propose a multistep photovoltaic power forecasting model that combines multi-timescale fluctuation aggregation with an attention mechanism and contrastive learning, demonstrating improved accuracy for renewable power series predic-

tion. Wang et al. [16] introduce a hybrid forecasting pipeline that integrates seasonal-trend decomposition with singular spectrum analysis and a meta-heuristic optimizer-embedded neural network, improving accuracy for volatile time-series scenarios. Additionally, Hong et al. [17] presents a correlational strategy that uses technical indicators and neural networks to address prediction accuracy in high-dimensional financial time series. While these studies highlight the benefits of decomposition or correlation strategies for forecasting, they mainly operate in supervised or hybrid learning settings and do not incorporate self-supervised contrastive representation learning. Therefore, it remains unclear whether component-wise representations learned in a self-supervised manner can improve downstream forecasting tasks.

### 2.3. Siamese Networks for Similarity Learning

Siamese Neural Networks (SNNs) [6,7] learn similarity functions between paired inputs through weight-sharing subnetworks. Originally used for verification tasks such as handwritten signatures and face matching [18], SNNs have since been adapted to time-series learning. SimTS [5] leverages a Siamese contrastive loss to remove negative-pair sampling and enhance robustness in time-series representation learning. However, SNN-based approaches do not incorporate decomposition or frequency-domain modeling and continue to operate on entangled raw signals.

### 2.4. Summary and Research Gap

In summary, contrastive learning methods improve representation learning for time series but rarely account for trend–season decomposition, forecasting objectives, or correlations across features. Similarly, decomposition- or correlation-based forecasting methods improve supervised forecasting but do not integrate self-supervised contrastive learning for representation learning. Existing Siamese contrastive approaches do not leverage component-wise structures or frequency features. These limitations motivate a decomposition-driven, forecasting-aware contrastive framework that learns component-wise representations in a self-supervised manner.

## 3. Preliminary

### 3.1. Problem Formulation

Given a time series $X = \{X_1, \cdots, X_M, \cdots, X_N\} \in \mathbb{R}^{C \times N}$, where $N$ is the length of the series and $X_t \in \mathbb{R}^C$ represents a $C$-dimensional data point at timestep $t$, with $C$ being the number of features or variables, we split the time series at point $M$ into two segments $X_{pre}$ and $X_{post}$:

$$X_{pre} = [X_1, \cdots, X_M] \in \mathbb{R}^{C \times M}, \; X_{post} = [X_{M+1}, \cdots, X_N] \in \mathbb{R}^{C \times (N-M)} \tag{1}$$

Here, $X_{pre}$ serves as the input to our model, while $X_{post}$ is the target that we aim to predict based on historical data. This setup defines our forecasting problem, where the goal is to predict $X_{post}$ using $X_{pre}$.

### 3.2. Time Series Decomposition

We further decompose both $X_{pre}$ and $X_{post}$ into trend $T$, seasonality $S$, and residual $R$ components using a sliding window method as follows:

$$X_t = T_t + S_t + R_t, \; 0 < t \leq N. \tag{2}$$

For $X_{pre}$, it is divided into $T_{pre}, S_{pre}, R_{pre} \in \mathbb{R}^{C \times M}$. Similarly, $X_{post}$ is divided into $T_{post}, S_{post}, R_{post} \in \mathbb{R}^{C \times (N-M)}$.

For clarity, we redefine the trend and seasonality components as follows: For trend segments, $\hat{T}_{pre} = [\hat{T}_1, \hat{T}_2, \cdots, \hat{T}_M]$ and $\hat{T}_{post} = [\hat{T}_{M+1}, \hat{T}_{M+2}, \cdots, \hat{T}_N]$. For seasonality segments, $\hat{S}_{pre} = [\hat{S}_1, \hat{S}_2, \cdots, \hat{S}_M]$ and $\hat{S}_{post} = [\hat{S}_{M+1}, \hat{S}_{M+2}, \cdots, \hat{S}_N]$.

### 3.3. Model Architecture

Our model incorporates the concept of Siamese neural networks, consisting of two identical subnetworks: the previous subnetwork and the target subnetwork. Each subnetwork has two encoder networks, denoted as $\mathcal{TD}_\theta$ and $\mathcal{SD}_\theta$, which are used to learn the trend representations $\mathcal{F}^{T,pre}$ and the seasonality representations $\mathcal{F}^{S,pre}$ . These representations are then used to predict the trend $\mathcal{F}^{T,post}$ and seasonality $\mathcal{F}^{S,post}$ in the target subnetwork.

Finally, the model is trained using contrastive learning, maximizing the similarity between the learned representations of $X_{pre}$ and $X_{post}$. This ensures that the model effectively captures meaningful patterns in time-series data, enhancing forecasting performance.

## 4. Methodologies

In this section, we present our proposed TSCL in detail. Figure 1 provides an overview of TSCL.

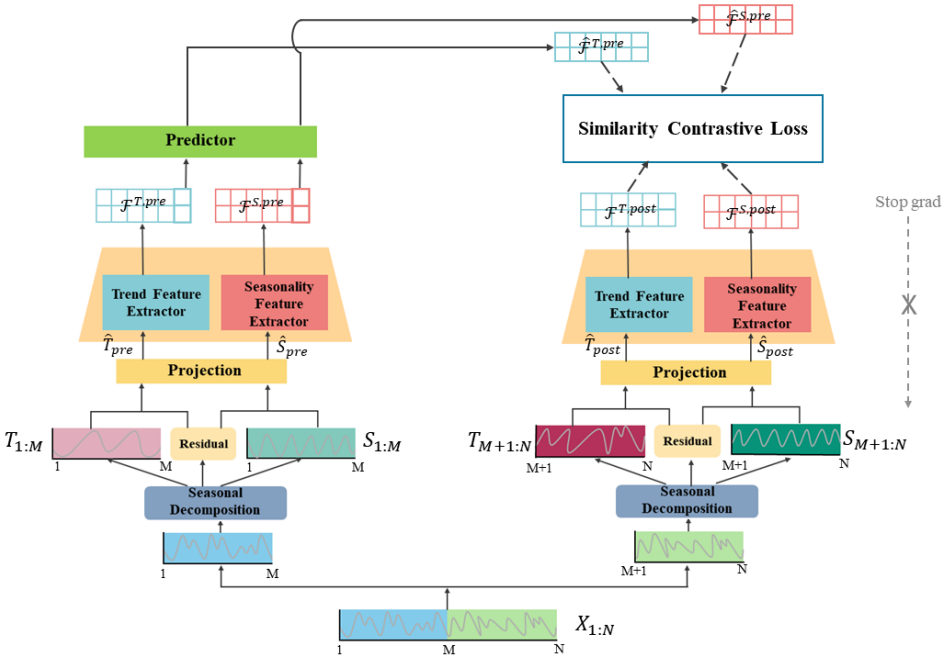



**Figure 1.** Architecture diagram of our proposed TSCL.

### 4.1. Seasonal Decomposition

To enhance forecasting performance, we decompose each time series into trend, seasonality, and residual components. The additive formulation is expressed as

$$X_t = T_t + S_t + R_t, \tag{3}$$

where $T_t$ captures low-frequency structure, $S_t$ captures periodic patterns, and $R_t$ denotes high-frequency residual fluctuations.

The trend component is estimated using a moving average with window size $k$:

$$T_t = \frac{1}{k} \sum_{i=t-k+1}^{t} X_i \tag{4}$$

The choice of $k = 25$ reflects a smoothing-scale hyperparameter that balances attenuation of short-term fluctuations with preservation of slow-varying structure. For the hourly ETT datasets, this corresponds to approximately one daily cycle, which is known to govern electricity load patterns. For the 15-min ETTm datasets, $k = 25$ spans approximately 6 h, which effectively removes transient variation while retaining diurnal behavior. From a spectral perspective, $k$ controls the cut-off between low and high frequency components, and similar values are used in classical decomposition methods to isolate seasonal periodicity. The detrended component is then obtained by:

$$\mathrm{deT}_t = X_t - T_t, \tag{5}$$

and the seasonal component is estimated by smoothing the detrended signal:

$$S_t = \mathrm{deT}_t - \frac{1}{k} \sum_{i=t-k+1}^{t} \mathrm{deT}_i \tag{6}$$

The residual term is defined as

$$R_t = X_t - T_t - S_t \tag{7}$$

For contrastive learning, we reallocate residuals into trend and seasonal representations:

$$\hat{T}_t = T_t + \beta R_t, \quad \hat{S}_t = S_t + (1 - \beta) R_t, \tag{8}$$

where $\beta$ determines the proportion of residual variation assigned to the trend. This formulation provides an interpretable mechanism for decomposing temporal information at different frequencies without requiring dataset-specific tuning.

### 4.2. Siamese Neural Network Architecture and Subnetwork Design

To enhance feature extraction, we employ a Siamese Neural Network (SNN) and Convolutional Neural Network (CNN) re-projection, enabling the model to learn relationships between trend and seasonality patterns more effectively. Inspired by SimTS [5], TSCL integrates Siamese architecture, consisting of two identical subnetworks: the previous and the target subnetwork.

Each subnetwork contains two encoder networks, a trend encoder $\mathcal{TD}_\theta$ and a seasonality encoder $\mathcal{SD}_\theta$, which process trend and seasonality components using shared parameters. The SNN learns a similarity metric by embedding inputs and measuring their similarity.

In our framework, the previous subnetwork extracts features from past trend and seasonality segments, while the target subnetwork processes corresponding target segments. To enhance high-dimensional feature mapping, we apply a CNN-based pre-processing module, improving the model's ability to capture complex temporal patterns.

### 4.3. Trend Feature Extractor

In TSCL, both the previous subnetwork and the target subnetwork contain a trend feature extractor $\mathcal{T}D_\theta$ with shared parameters. As shown in Figure 2, the trend feature extractor employs multiple 1-dimensional causal convolution layers with kernel sizes $2^i$ for $i = 0, 1, \ldots, k$, to capture trend representations across different time scales by various kernel sizes. The outputs from these convolution layers are aggregated using an average pooling layer, enabling the model to effectively learn both short-term and long-term dependencies.

By incorporating causal convolution layers with diverse kernel sizes, $\mathcal{T}D_\theta$ effectively captures intricate temporal patterns in the data. Smaller kernels focus on fine-grained, short-

term trends, while larger kernels are suited for identifying broader, long-term patterns. This multi-scale approach ensures a more comprehensive understanding of temporal dynamics.

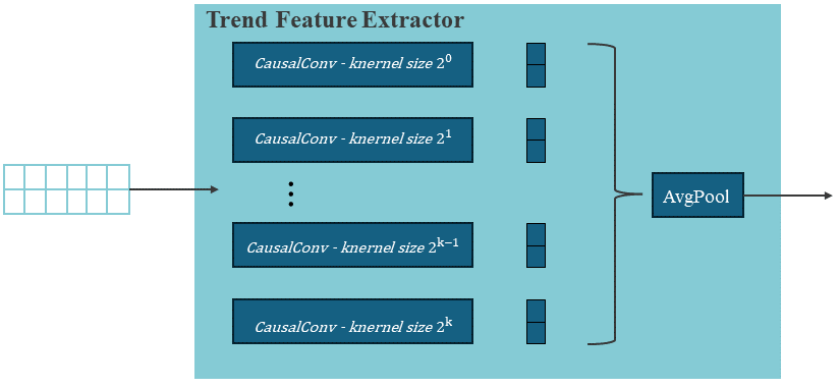



**Figure 2.** Architecture of the Trend Feature Extractor.

As described in Section 4.2, our model adopts a Siamese Neural Network structure with two identical subnetworks: the previous subnetwork and the target subnetwork. Time-series data is segmented at time point $M$. After processing through the Seasonal Decomposition module, we obtain the previous trend time-series segments $\hat{T}_{pre}$ and the target trend time-series segments $\hat{T}_{post}$ as follows:

$$\hat{T}_{pre} = [\hat{T}_1, \hat{T}_2, \cdots, \hat{T}_M], \hat{T}_{post} = [\hat{T}_{M+1}, \hat{T}_{M+2}, \cdots, \hat{T}_N] \tag{9}$$

These segments are then passed through the trend feature extractors $\mathcal{TD}_\theta$ in both subnetworks to extract trend feature representations $\mathcal{F}^{T,pre}$ and $\mathcal{F}^{T,post}$ as follows:

$$\mathcal{F}^{T,pre} = \mathcal{TD}_\theta(\hat{T}_{pre}) = \left[\mathcal{F}_1^{T,pre}, \mathcal{F}_2^{T,pre}, \cdots, \mathcal{F}_M^{T,pre}\right] \tag{10}$$

$$\mathcal{F}^{T,post} = \mathcal{TD}_\theta(\hat{T}_{post}) = \left[\mathcal{F}_{M+1}^{T,post}, \mathcal{F}_{M+2}^{T,post}, \cdots, \mathcal{F}_N^{T,post}\right] \tag{11}$$

To improve generalization and prevent overfitting, we incorporate the stop-gradient mechanism inspired by SimSiam [6,7]. This mechanism ensures that when sharing parameters between the previous and target subnetworks, the trend feature extractor does not update its parameters based on target trend segments. This encourage the model to learn robust, generalized patterns from previous segments instead of overfitting to the target data.

### *4.4. Seasonality Feature Extractor*

Spectral analysis in the frequency domain has been extensively utilized for detecting seasonality. Inspired by CoST [4], we also employ frequency domain analysis to capture and analyze seasonal patterns in time-series data. In our model, both the previous subnetwork and the target subnetwork contain a seasonality feature extractor, denoted as $\mathcal{SD}_\theta$, which shares parameters across both subnetworks. As shown in Figure 3, the seasonality feature extractor applies the Fourier Transform to decompose time-series data into frequency components, allowing for a more precise identification of seasonal patterns.

The core component of the seasonality feature extractor is the Banded Fourier Layer. This layer is designed to isolate specific frequency bands within the Fourier-transformed signals. This allows for targeted analysis of different periodic components, enhancing the model's ability to extract meaningful seasonality patterns.

The Banded Fourier Layer begins by applying a Discrete Fourier Transform (DFT) along the time dimension of the intermediate feature representations. This transformation

converts the time domain data into the frequency domain, resulting in a complex-valued matrix. A learnable layer then applies a linear matrix transformation to each frequency component using unique complex-valued parameters, enabling frequency domain interactions. The transformed frequency components are then converted back to the time domain using an inverse DFT operation, resulting in time-domain signals.

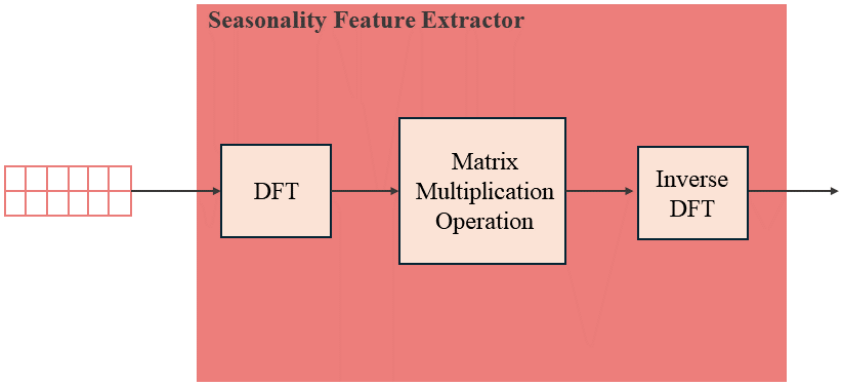



**Figure 3.** Architecture of the Seasonality Feature Extractor.

Similar to the trend feature extractor, the seasonality feature extractor also follows the SNN framework. The time-series data is segmented at time point $M$. After processing through the Seasonal Decomposition module, we obtain the previous seasonality time-series segments $\hat{S}_{pre}$ and the target seasonality time-series segments $\hat{S}_{post}$ as follows:

$$\hat{S}_{pre} = [\hat{S}_1, \hat{S}_2, \cdots, \hat{S}_M],\ \hat{S}_{post} = [\hat{S}_{M+1}, \hat{S}_{M+2}, \cdots, \hat{S}_N] \tag{12}$$

These segments are then processed through the seasonality feature extractors $\mathcal{SD}_\theta$ in both subnetworks to obtain seasonality feature representations $\mathcal{F}^{S,pre}$ and $\mathcal{F}^{S,post}$ as follows:

$$\mathcal{F}^{S,pre} = \mathcal{SD}_\theta(\hat{S}_{pre}) = \left[\mathcal{F}_1^{S,pre}, \mathcal{F}_2^{S,pre}, \cdots, \mathcal{F}_M^{S,pre}\right] \tag{13}$$

$$\mathcal{F}^{S,post} = \mathcal{SD}_\theta(\hat{S}_{post}) = \left[\mathcal{F}_{M+1}^{S,post}, \mathcal{F}_{M+2}^{S,post}, \cdots, \mathcal{F}_N^{S,post}\right] \tag{14}$$

These representations help capture periodic structures, improving the model's ability to learn meaningful seasonality patterns for forecasting.

### 4.5. Predictor

As proposed by [19,20], we utilize the last row of the feature representations $\mathcal{F}_M^{T,pre}$ and $\mathcal{F}_M^{S,pre}$ obtained from the previous subnetwork to predict the corresponding representations in the target subnetwork. After obtaining $\mathcal{F}_M^{T,pre}$ and $\mathcal{F}_M^{S,pre}$, these feature representations are then passed into a Multilayer Perceptron (MLP) predictor $\mathcal{P}_\theta$ to estimate the feature representations of the target subnetwork, denoted as $\hat{\mathcal{F}}^{T,pre}$ and $\hat{\mathcal{F}}^{S,pre}$. The MLP predictor $\mathcal{P}_\theta$ is defined as follows:

$$\mathcal{P}_\theta(x) = \mathrm{Wk}_{w12}(\mathrm{ReLU}(\mathrm{Wk}_{w1}(x))^T), \tag{15}$$

where $\mathrm{Wk}_{w1}$ represents a linear transformation that maps input features of input lengths to a feature space of output lengths, and $\mathrm{Wk}_{w12}$ is another linear transformation that maps the data, after ReLU activation, to a feature space of output dimensions. The feature representations can be predicted using the following equations:

$$\hat{\mathcal{F}}^{T,pre} = \mathcal{P}_\theta(\mathcal{F}_M^{T,pre}), \quad \hat{\mathcal{F}}^{S,pre} = \mathcal{P}_\theta(\mathcal{F}_M^{S,pre}) \tag{16}$$

We define the predicted target trend feature representations $\hat{\mathcal{F}}^{T,pre}$ from the predictor and the actual target subnetwork trend feature representations $\mathcal{F}^{T,post}$ as a positive pair. Similarly, the predicted target seasonality feature representations $\hat{\mathcal{F}}^{S,pre}$ from the predictor and the actual target subnetwork seasonality feature representations $\mathcal{F}^{S,post}$ also form a positive pair. In contrastive learning, we expect that positive pairs are as similar as possible.

Using this structure, the predictor learns to generate feature representations that align with those of the target subnetwork, aiding in robust contrastive learning.

### 4.6. Similarity Contrastive Loss

To compute the loss, we combine the trend cosine similarity loss $L_{trend}$ and the seasonality cosine similarity loss $L_{seasonality}$, capturing the trend and seasonality spectral features of the time-series data.

First, we convert the predicted target seasonality feature representations $\hat{\mathcal{F}}^{S,pre}$ and the target seasonality feature representations $\mathcal{F}^{S,post}$ into their amplitude and phase components. This conversion allows us to analyze the spectral characteristics of seasonality features. The seasonality cosine similarity loss $L_{seasonality}$ measures the negative cosine similarity between positive pairs. We concatenate the feature representations and compute the pairwise cosine similarity.

For the seasonality cosine similarity loss, we first compute the amplitude cosine similarity:

$$\cos_sim_{\text{amp}} = \frac{1}{N-M}\sum_{i=M+1}^{N}\frac{\text{amp}(\text{FFT}(\hat{F}_i^{S,pre}))\cdot\text{amp}(\text{FFT}(F_i^{S,post}))}{\|\text{amp}(\text{FFT}(\hat{F}_i^{S,pre}))\|\|\text{amp}(\text{FFT}(F_i^{S,post}))\|} \tag{17}$$

Next, we compute the phase cosine similarity:

$$\cos_sim_{\text{ph}} = \frac{1}{N-M}\sum_{i=M+1}^{N}\frac{\text{phase}(\text{FFT}(\hat{F}_i^{S,pre}))\cdot\text{phase}(\text{FFT}(F_i^{S,post}))}{\|\text{phase}(\text{FFT}(\hat{F}_i^{S,pre}))\|\|\text{phase}(\text{FFT}(F_i^{S,post}))\|} \tag{18}$$

Then, the seasonality cosine similarity loss $L_{seasonality}$ is given by:

$$L_{seasonality} = -\frac{\cos_sim_{\text{amp}} + \cos_sim_{\text{ph}}}{2} \tag{19}$$

The trend cosine similarity loss $L_{trend}$ calculates the negative cosine similarity between the predicted trend feature representations $\hat{F}^{T,pre}$ and the target trend feature representations $F^{T,post}$:

$$L_{trend} = -\frac{1}{N-M}\sum_{i=M+1}^{N}\frac{\hat{F}_i^{T,pre}\cdot F_i^{T,post}}{\|\hat{F}_i^{T,pre}\|\|F_i^{T,post}\|} \tag{20}$$

The overall loss function $L$ is a weighted combination of the trend cosine similarity loss $L_{trend}$ and the seasonality cosine similarity loss $L_{seasonality}$, controlled by the hyperparameter $\alpha$ as follows:

$$L = L_{trend} + \alpha\cdot L_{seasonality} \tag{21}$$

## 5. Experiment

### 5.1. Dataset

We evaluated TSCL using five real-world public benchmark datasets, as shown in Table 1. The Electricity Transformer Temperature (ETT) dataset [21] consists of four subsets: two sampled hourly (ETTh) and two sampled every 15 min (ETTm). These datasets measure six types of electricity load features and oil temperature, collected over two years from two provinces in China. The Weather dataset contains hourly local climate data from nearly

1600 regions across the United States over four years, collected every hour, with each time step containing 12 weather variables.

**Table 1.** Summary of five real-world public benchmark datasets.

| Dataset | Variable Number | Frequency | Aggregate Number of Observations |
|---|---|---|---|
| ETTh1 | 7 | 1 h | 17,420 |
| ETTh2 | 7 | 1 h | 17,420 |
| ETTm1 | 7 | 15 min | 69,680 |
| ETTm2 | 7 | 15 min | 69,680 |
| Weather | 12 | 1 h | 35,064 |

### *5.2. Experiment Settings*

We split all datasets into training, validation, and test sets using a 6:2:2 ratio. Model evaluation was performed using mean square error (MSE) and mean absolute error (MAE) as metrics.

To decompose the time-series data, we used a seasonal model with a window size of 25, adjusting the proportion of residuals in trend and seasonality through the hyperparameter $\beta$, fine-tuned experimentally. For the ETT datasets, we set $\beta$ to 0.2 and $\alpha$ to 1.0 for hourly data; for ETTm1, we set $\beta$ to 0.4 and $\alpha$ to 0.6; and for ETTm2, we set $\beta$ to 0.4 and $\alpha$ to 0.8. For the Weather dataset, we used $\beta = 0.5$ and $\alpha = 0.1$. The trend and seasonal components were projected into a 64-dimensional latent space using a convolutional projector. Additionally, trend sequences were encoded into a 320-dimensional space using multi-scale convolution. Each training sample consisted of subsequences of length 402, with the first 201 timestamps as input and the next 201 as the target trend segment. We optimized cosine similarity loss using the stochastic gradient descent (SGD) with a learning rate of 0.001, momentum of 0.9, and weight decay of 0.0001.

For the ETTh1, ETTh2, and Weather datasets, prediction horizons were set to $L \in \{24, 48, 168, 336, 720\}$, while for the ETTm1 and ETTm2 datasets, these were set to $L \in \{24, 48, 96, 288, 672\}$. The optimal ridge regression model was identified on the validation set and applied for error measurement.

### *5.3. Baselines*

We compared TSCL with various two-stage representation learning methods for time series, including SimTS [5], CoST [4], TS2Vec [2], and TNC [9]. We also employed several end-to-end models, including Informer [21] and TCN [22].

### *5.4. Overall Performance*

Table 2 reports the multivariate forecasting results across five datasets and multiple prediction horizons. TSCL achieves the best average performance in both MSE and MAE, outperforming SimTS, CoST, TS2Vec, and TNC as well as end-to-end forecasting baselines such as Informer and TCN. The improvements are particularly consistent on ETTh1, ETTh2, and Weather, and at medium to long forecasting horizons where seasonal structure contributes more strongly to predictive performance.

The results are not fully uniform across all experimental settings. At the shortest forecasting horizon $L = 24$, TSCL ranks second on ETTm1, ETTm2, and Weather. In these three cases, SimTS achieves the lowest error, and CoST performs equally well as SimTS on Weather at $L = 24$. These observations indicate that short-horizon forecasting, especially for high-frequency series such as ETTm1 and ETTm2, benefits more from objectives that emphasize local temporal similarity rather than seasonal decomposition.

**Table 2.** The results of multivariate forecasting. The best result in each row is in boldface and the second place is underlined.

| Methods | | Unsupervised Representation Learning | | | | | | | | | | End-to-End Forecasting | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | TSCL | | SimTS | | CoST | | TS2Vec | | TNC | | Informer | | TCN | |
| L | | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE |
| ETTh1 | 24 | **0.309** | **0.382** | 0.377 | 0.422 | 0.386 | 0.429 | 0.590 | 0.531 | 0.708 | 0.592 | 0.577 | 0.549 | 0.583 | 0.547 |
| | 48 | **0.331** | **0.392** | 0.427 | 0.454 | 0.437 | 0.464 | 0.624 | 0.555 | 0.749 | 0.619 | 0.685 | 0.625 | 0.670 | 0.606 |
| | 168 | **0.421** | **0.452** | 0.638 | 0.577 | 0.643 | 0.582 | 0.762 | 0.639 | 0.884 | 0.699 | 0.931 | 0.752 | 0.811 | 0.680 |
| | 336 | **0.692** | **0.593** | 0.815 | 0.685 | 0.812 | 0.679 | 0.931 | 0.728 | 1.020 | 0.768 | 1.128 | 0.873 | 1.132 | 0.815 |
| | 720 | **0.848** | **0.724** | 0.956 | 0.771 | 0.970 | 0.771 | 1.063 | 0.799 | 1.157 | 0.830 | 1.215 | 1.869 | 1.165 | 0.813 |
| ETTh2 | 24 | **0.29** | **0.443** | 0.336 | 0.434 | 0.447 | 0.502 | 0.424 | 0.489 | 0.612 | 0.595 | 0.720 | 0.665 | 0.935 | 0.754 |
| | 48 | **0.387** | **0.490** | 0.564 | 0.571 | 0.699 | 0.637 | 0.619 | 0.605 | 0.84 | 0.716 | 1.457 | 1.001 | 1.300 | 0.911 |
| | 168 | **0.719** | **0.658** | 1.407 | 0.926 | 1.549 | 0.982 | 1.845 | 1.074 | 2.359 | 1.213 | 3.489 | 1.515 | 4.017 | 1.579 |
| | 336 | **1.265** | **0.874** | 1.640 | 0.996 | 1.749 | 1.042 | 2.194 | 1.197 | 2.782 | 1.349 | 2.723 | 1.340 | 3.460 | 1.456 |
| | 720 | **1.582** | **0.991** | 1.878 | 1.065 | 1.971 | 1.092 | 2.636 | 1.37 | 2.753 | 1.394 | 3.467 | 1.376 | 3.106 | 1.381 |
| ETTm1 | 24 | **0.217** | 0.317 | 0.232 | **0.314** | 0.246 | 0.329 | 0.453 | 0.444 | 0.522 | 0.472 | 0.323 | 0.369 | 0.522 | 0.472 |
| | 48 | **0.267** | **0.350** | 0.311 | 0.368 | 0.381 | 0.386 | 0.592 | 0.521 | 0.695 | 0.567 | 0.494 | 0.503 | 0.542 | 0.508 |
| | 96 | **0.295** | **0.366** | 0.360 | 0.402 | 0.378 | 0.419 | 0.635 | 0.554 | 0.731 | 0.595 | 0.678 | 0.614 | 0.666 | 0.578 |
| | 288 | **0.342** | **0.399** | 0.450 | 0.467 | 0.472 | 0.486 | 0.693 | 0.597 | 0.818 | 0.649 | 1.056 | 0.786 | 0.991 | 0.735 |
| | 672 | **0.425** | **0.445** | 0.612 | 0.563 | 0.620 | 0.574 | 0.782 | 0.653 | 0.932 | 0.712 | 1.192 | 0.926 | 1.032 | 0.756 |
| ETTm2 | 24 | 0.135 | 0.269 | **0.108** | **0.223** | 0.122 | 0.244 | 0.180 | 0.293 | 0.185 | 0.297 | 0.173 | 0.301 | 0.180 | 0.324 |
| | 48 | **0.153** | **0.283** | 0.164 | 0.285 | 0.183 | 0.305 | 0.244 | 0.350 | 0.264 | 0.36 | 0.303 | 0.409 | 0.204 | 0.327 |
| | 96 | **0.183** | **0.311** | 0.271 | 0.376 | 0.294 | 0.394 | 0.360 | 0.427 | 0.389 | 0.458 | 0.365 | 0.453 | 3.041 | 1.330 |
| | 288 | **0.459** | **0.516** | 0.716 | 0.646 | 0.723 | 0.652 | 0.723 | 0.639 | 0.920 | 0.788 | 1.047 | 0.804 | 3.162 | 1.337 |
| | 672 | **0.899** | **0.722** | 1.600 | 0.979 | 1.899 | 1.073 | 1.753 | 1.007 | 2.164 | 1.135 | 3.126 | 1.302 | 3.624 | 1.484 |
| Weather | 24 | 0.318 | 0.371 | **0.298** | **0.359** | **0.298** | 0.360 | 0.308 | 0.364 | 0.320 | 0.373 | 0.335 | 0.381 | 0.321 | 0.367 |
| | 48 | **0.359** | **0.408** | 0.359 | 0.410 | 0.359 | 0.411 | 0.375 | 0.417 | 0.380 | 0.421 | 0.395 | 0.459 | 0.386 | 0.423 |
| | 168 | **0.416** | **0.456** | 0.426 | 0.461 | 0.464 | 0.491 | 0.496 | 0.506 | 0.479 | 0.495 | 0.608 | 0.567 | 0.491 | 0.501 |
| | 336 | **0.428** | **0.470** | 0.504 | 0.520 | 0.497 | 0.517 | 0.532 | 0.533 | 0.505 | 0.514 | 0.702 | 0.620 | 0.502 | 0.507 |
| | 720 | **0.491** | **0.514** | 0.535 | 0.542 | 0.533 | 0.542 | 0.567 | 0.558 | 0.543 | 0.547 | 0.831 | 0.731 | 0.498 | 0.508 |
| Average | | **0.489** | **0.488** | 0.639 | 0.553 | 0.685 | 0.575 | 0.815 | 0.634 | 0.948 | 0.686 | 1.121 | 0.792 | 1.334 | 0.788 |

It is noteworthy that all three instances where TSCL ranks second correspond to the shortest horizon of $L = 24$. This pattern suggests that fine-scale temporal invariances are more influential for near-term prediction, whereas seasonal decomposition becomes more advantageous as the horizon increases and long-range periodic structure accumulates. Despite this nuance, TSCL remains competitive in these settings and attains the best aggregated performance across datasets and horizons, demonstrating that seasonally informed representations provide robust generalization for multivariate forecasting.

Beyond the numerical differences in forecasting error, we assess whether the improvements of TSCL are statistically significant. We perform paired $t$-tests (df = 4) comparing TSCL against SimTS and CoST, the two strongest unsupervised baselines. As shown in Table 3, TSCL achieves statistical significance in 19/25 cases against SimTS and 21/25 cases against CoST for MSE, and in 19/25 and 23/25 cases for MAE, respectively.

The few non-significant cases concentrate primarily at the shortest forecasting horizon on ETTm2 and Weather. For ETTm2, the absence of significance aligns with the high-frequency nature (15-min sampling) of the dataset, where local temporal similarity plays a larger role than seasonal decomposition. For Weather, although its sampling frequency matches ETTh1 and ETTh2 (hourly), meteorological multivariate series exhibit weaker periodicity and higher noise, making seasonal signals less dominant at short horizons. These observations are consistent with the results in Table 2, where TSCL exhibits increasing advantages at medium and long horizons across all datasets.

Taken together, these findings indicate that TSCL's seasonal decomposition mechanism yields statistically reliable improvements, particularly when forecasting beyond short-term horizons where periodic structure accumulates.

**Table 3.** Paired $t$-test (df = 4) results comparing TSCL against SimTS and CoST on multivariate forecasting. $p$-values are obtained from paired $t$-tests over five independent runs with different random seeds. Each cell reports $p$-value/significance indicator (✓ for $p < 0.05$, × for $p \geq 0.05$).

| Dataset | $L$ | MSE (TSCL vs. Baseline) | | MAE (TSCL vs. Baseline) | |
|---|---|---|---|---|---|
| | | SimTS | CoST | SimTS | CoST |
| ETTh1 | 24 | 0.000045/✓ | 0.000305/✓ | 0.000354/✓ | 0.001495/✓ |
| | 48 | 0.000015/✓ | 0.000099/✓ | 0.000083/✓ | 0.000418/✓ |
| | 168 | 0.000004/✓ | 0.000037/✓ | 0.000011/✓ | 0.000037/✓ |
| | 336 | 0.000390/✓ | 0.000213/✓ | 0.000092/✓ | 0.000034/✓ |
| | 720 | 0.053035/× | 0.061625/× | 0.014611/✓ | 0.030061/✓ |
| ETTh2 | 24 | 0.003071/✓ | 0.000978/✓ | 0.096234/× | 0.007458/✓ |
| | 48 | 0.000293/✓ | 0.000040/✓ | 0.000641/✓ | 0.000067/✓ |
| | 168 | 0.000319/✓ | 0.000359/✓ | 0.000671/✓ | 0.000469/✓ |
| | 336 | 0.000375/✓ | 0.000429/✓ | 0.003370/✓ | 0.002133/✓ |
| | 720 | 0.000116/✓ | 0.000121/✓ | 0.002703/✓ | 0.006294/✓ |
| ETTm1 | 24 | 0.007458/✓ | 0.000243/✓ | 0.404433/× | 0.005564/✓ |
| | 48 | 0.000068/✓ | 0.000001/✓ | 0.001506/✓ | 0.000034/✓ |
| | 96 | 0.000016/✓ | 0.000001/✓ | 0.000071/✓ | 0.000003/✓ |
| | 288 | 0.000001/✓ | 0.000034/✓ | 0.000012/✓ | 0.000078/✓ |
| | 672 | 0.000001/✓ | 0.000047/✓ | 0.000002/✓ | 0.000044/✓ |
| ETTm2 | 24 | 0.023821/✓ | 0.337973/× | 0.001518/✓ | 0.092175/× |
| | 48 | 0.072397/× | 0.059594/× | 0.747555/× | 0.139141/× |
| | 96 | 0.000102/✓ | 0.000231/✓ | 0.000392/✓ | 0.000118/✓ |
| | 288 | 0.000085/✓ | 0.000283/✓ | 0.000127/✓ | 0.000052/✓ |
| | 672 | 0.003921/✓ | 0.001070/✓ | 0.001671/✓ | 0.000690/✓ |
| Weather | 24 | 0.254438/× | 0.000001/✓ | 0.320377/× | 0.000004/✓ |
| | 48 | 0.986316/× | 0.373901/× | 0.869525/× | 0.000000/✓ |
| | 168 | 0.056207/× | 0.000000/✓ | 0.013803/✓ | 0.000003/✓ |
| | 336 | 0.013948/✓ | 0.000003/✓ | 0.001932/✓ | 0.000004/✓ |
| | 720 | 0.389858/× | 0.000022/✓ | 0.098605/× | 0.000028/✓ |
| Summary (Sig/Total) | | 19/25 | 21/25 | 19/25 | 23/25 |

### *5.5. Ablation Study*

To evaluate the impact of each component within our framework, we compare the performance of the model with the following: (1) **w/o SFE**: Full TSCL removes the Seasonality Feature Extractor (SFE) module; (2) **w/o SD**: Full TSCL removes the Seasonal Decomposition (SD) component; (3) **w/o SG**: Full TSCL removes the stop-gradient (SG) operation; (4) **w/ NPs**: Full TSCL incorporates negative pairs (NPs) into the model.

The results in Table 4 show that TSCL performs better on all datasets, highlighting the importance of each component in achieving optimal model performances. We demonstrate that removing either the Seasonal Decomposition or the Seasonality Feature Extractor decreases performance. We verify that effective training on both trend and seasonal data improves performance. The experimental results indicate that removing the stop-gradient operation further degrades model performance. Thus, the stop-gradient operation is crucial for stabilizing the training process and improving model performance. Regarding negative-pair construction, we follow the methodology of [23,24] to demonstrate that TSCL can operate with or without negative pairs. These results evaluate the architectural component rather than general conclusions about negative-pair strategies.

**Table 4.** The results of the ablation study. The best result is in boldface.

| | w/o SFE | | w/o SD | | w/o SG | | w/ NPs | | TSCL | |
|---|---|---|---|---|---|---|---|---|---|---|
| | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE |
| ETTh1 | 0.605 | 0.540 | 0.770 | 0.654 | 0.627 | 0.560 | 0.567 | 0.531 | **0.520** | **0.509** |
| ETTh2 | 0.962 | 0.732 | 1.215 | 0.809 | 1.616 | 0.941 | 0.949 | 0.726 | **0.849** | **0.691** |
| ETTm1 | 0.322 | 0.382 | 0.420 | 0.446 | 0.358 | 0.427 | 0.319 | 0.381 | **0.309** | **0.375** |
| ETTm2 | 0.497 | 0.468 | 0.744 | 0.567 | 0.807 | 0.668 | 0.411 | 0.435 | **0.366** | **0.420** |
| Weather | 0.404 | 0.445 | 0.429 | 0.463 | 0.437 | 0.473 | 0.403 | 0.445 | **0.402** | **0.444** |

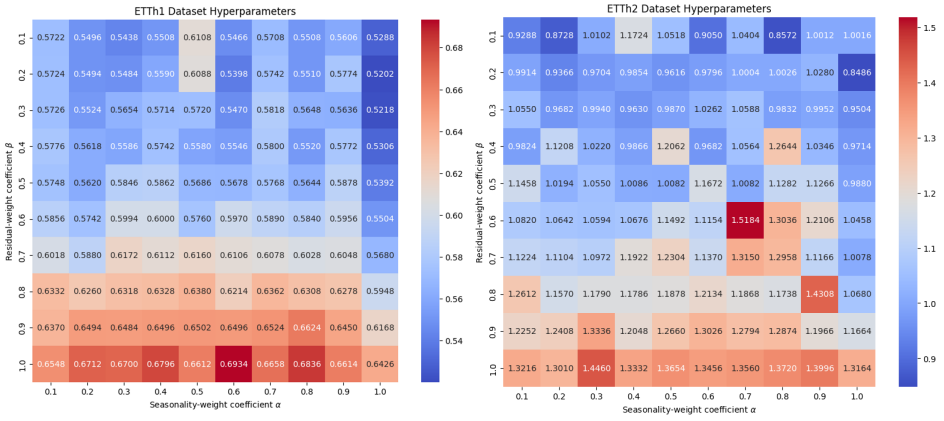
### *5.6. Hyperparameter Study*

In this section, we analyze the effect of the two key hyperparameters $\alpha$ and $\beta$ using the ETT and Weather datasets.

The parameter $\alpha$ balances the trend-oriented and seasonality-oriented objectives in the loss function. This weighting can be viewed as a scalarization of a bi-objective optimization problem, where the model must simultaneously capture low-frequency trend structure and high-frequency seasonal structure. Without such a balancing mechanism, the two branches contribute gradients of different scales, which can bias optimization toward either trend-only or seasonality-only representations. From a spectral perspective, $\alpha$ modulates the relative emphasis on low-frequency versus high-frequency components.

The hyperparameter $\beta$ controls the allocation of residual components during Seasonal Decomposition by adjusting the proportion of noise assigned to trend versus seasonality. Larger $\beta$ increases residual contribution to the trend, while $(1-\beta)$ increases residual contribution to seasonality. This mechanism allows the decomposition to adapt to datasets with different sampling rates and variance characteristics. To examine how these hyperparameters interact, we evaluate all combinations of $\alpha$ and $\beta$ within the range $[0.1, 1.0]$ in increments of 0.1.

Figure 4 shows the results for ETTh1 and ETTh2, both hourly electricity datasets with pronounced seasonal patterns. The optimal configuration occurs at $\alpha = 1.0$ and $\beta = 0.2$, indicating that emphasizing seasonality while allocating minimal residual noise to the trend yields the best performance. This observation aligns with the fact that ETTh1 and ETTh2 exhibit strong periodic behavior, making seasonal structure more informative than residual short-term fluctuations.

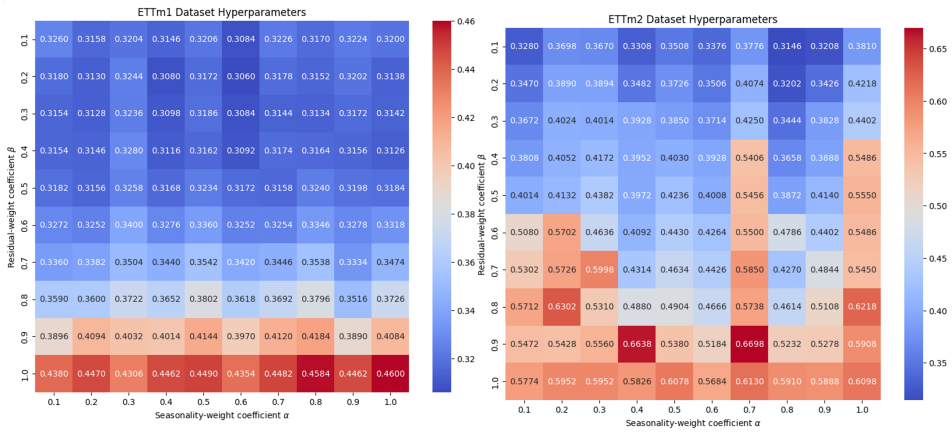



(**a**) (**b**)

**Figure 4.** Heatmaps of (**a**) ETTh1 and (**b**) ETTh2 showing forecasting error across combinations of $\alpha$ and $\beta$. Values represent MSE averaged across prediction horizons (lower is better). Colors correspond to error magnitude, with cooler colors indicating lower error and warmer colors indicating higher error; the colorbar denotes the quantitative scale.

Figure 5 illustrates the effect on ETTm1 and ETTm2, which are recorded at 15-min intervals and contain more high-frequency variation. The best performance occurs at $(\alpha, \beta) = (0.6, 0.2)$ for ETTm1 and $(\alpha, \beta) = (0.8, 0.1)$ for ETTm2, suggesting that a higher residual contribution and moderate seasonal emphasis are beneficial for high-frequency data. This confirms the optimization viewpoint that $\alpha$ governs the trade-off between fine-scale and periodic information, while $\beta$ adapts decomposition to the underlying noise characteristics.

Finally, Figure 6 presents results for the Weather dataset, which contains meteorological variables exhibiting complex variability and irregular fluctuations. The optimal configuration is $(\alpha, \beta) = (0.1, 0.3)$, indicating that limited seasonal emphasis and higher residual allocation are more effective. This is consistent with the less reliable periodic structure in meteorological time series and the presence of abrupt short-term changes.




(**a**) (**b**)

**Figure 5.** Heatmaps of (**a**) ETTm1 and (**b**) ETTm2 showing forecasting error across combinations of $\alpha$ and $\beta$. Values represent MSE averaged across prediction horizons (lower is better). Colors correspond to error magnitude, with cooler colors indicating lower error and warmer colors indicating higher error; the colorbar denotes the quantitative scale.

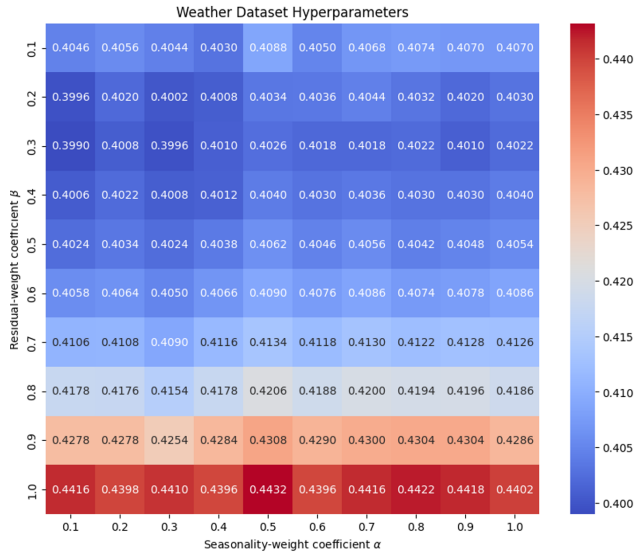



**Figure 6.** Heatmap of Weather dataset showing forecasting error across combinations of $\alpha$ and $\beta$. Values represent MSE averaged across prediction horizons (lower is better). Colors correspond to error magnitude, with cooler colors indicating lower error and warmer colors indicating higher error; the colorbar denotes the quantitative scale.

Taken together, these results validate that $\alpha$ and $\beta$ encode interpretable trade-offs rather than heuristic tunings. The former corresponds to multi-objective balancing in the spectral domain, while the latter adjusts decomposition noise allocation. Different datasets favor different settings depending on sampling rate, periodicity, and variability, supporting the motivation for a flexible weighted decomposition design.

## 6. Limitations and Future Work

While TSCL achieves promising forecasting performance across benchmark datasets, several limitations and opportunities for further research should be noted. First, the empirical results indicate that TSCL benefits from datasets exhibiting clear seasonal or trend structure, where decomposition enhances feature learning. In contrast, highly irregular or sparse domains may require additional mechanisms for handling missingness, temporal alignment, or abrupt regime changes.

Second, our evaluation focuses on regularly sampled multivariate time series from the energy and meteorological domains. Many real-world applications in healthcare, finance, and telemetry involve irregular sampling, sparse observations, and cross-variable dependencies. Extending TSCL to such settings represents an important direction for increasing its practical applicability.

Third, TSCL is introduced as a self-supervised representation learning framework rather than a new forecasting backbone. Although the experimental evaluation includes representative contrastive learning baselines and end-to-end forecasting models, a broader comparison against the expanding family of Transformer-based forecasting architectures (e.g., Autoformer, FEDFormer) would provide a more comprehensive performance landscape. Conducting such comparisons or integrating TSCL representations with these backbones constitutes a natural direction for future work.

Fourth, the decomposition step assumes additive interactions between trend and seasonal components. Although standard in classical time-series analysis, nonlinear seasonal behavior and structural breaks may require adaptive or learnable decomposition strategies.

Fifth, TSCL focuses on point forecasting accuracy. Many operational applications benefit from uncertainty quantification or probabilistic forecasting; incorporating such objectives remains for future work.

Finally, while TSCL removes negative-pair construction through a Siamese objective, the broader relationship between negative-pair strategies and downstream generalization remains task-dependent and not empirically settled. A systematic comparison of negative-pair formulations constitutes a promising direction for future investigation.

Decomposition-based contrastive learning may be particularly useful in domains characterized by periodic or quasi-periodic behavior, including energy demand, environmental monitoring, and transportation forecasting. Overall, these limitations highlight opportunities to broaden the robustness, scope, and real-world applicability of TSCL in future research.

## 7. Conclusions

This paper presents the **T**rend-**S**eason **C**ontrastive **L**earner (TSCL), a decomposition-based self-supervised framework for multivariate time-series forecasting. By leveraging seasonal decomposition, a learnable Fourier layer, and a Siamese contrastive objective, TSCL learns seasonally informed representations that benefit downstream forecasting tasks.

Extensive experiments on five real-world datasets demonstrate that TSCL improves forecasting accuracy compared to existing contrastive learning methods and end-to-end forecasting models. In terms of aggregated performance across datasets and forecasting horizons, TSCL achieves the lowest average error (MSE = 0.489, MAE = 0.488), outper-

forming SimTS (MSE = 0.639, MAE = 0.553), CoST (MSE = 0.685, MAE = 0.575), TS2Vec (MSE = 0.815, MAE = 0.634), and TNC (MSE = 0.948, MAE = 0.686), as well as end-to-end baselines such as Informer and TCN. These results indicate that seasonally structured representations contribute to more robust forecasting performance, particularly at medium and long prediction horizons.

Together, the decomposition and contrastive learning components provide a principled approach to capturing both periodic and slowly varying patterns in multivariate time series. The findings suggest that decomposition-based self-supervised learning is a promising direction for representation learning in forecasting applications.

**Author Contributions:** Supervision, H.-P.H.; methodology, H.-P.H., Y.-C.L. and C.-R.C.; validation, C.-R.C., Y.-C.L. and P.-X.L.; writing—original draft preparation, C.-R.C. and Y.-C.L.; writing—review and editing, H.-P.H. and P.-X.L. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was funded by National Science and Technology Council (NSTC) of Taiwan under Grants 114-2628-E-006-005-MY3 and 112-2221-E-006-150-MY3.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** The datasets used in this study are publicly available benchmark datasets (ETTh1, ETTh2, ETTm1, ETTm2, and Weather), which can be obtained from their official repositories. Further inquiries can be directed to the corresponding author.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Ayala Solares, J.R.; Diletta Raimondi, F.E.; Zhu, Y.; Rahimian, F.; Canoy, D.; Tran, J.; Pinho Gomes, A.C.; Payberah, A.H.; Zottoli, M.; Nazarzadeh, M.; et al. Deep learning for electronic health records: A comparative review of multiple deep neural architectures. *J. Biomed. Inform.* **2020**, *101*, 103337. [CrossRef] [PubMed]
2. Yue, Z.; Wang, Y.; Duan, J.; Yang, T.; Huang, C.; Tong, Y.; Xu, B. TS2Vec: Towards Universal Representation of Time Series. *arXiv* **2022**, arXiv:2106.10466. [CrossRef]
3. Eldele, E.; Ragab, M.; Chen, Z.; Wu, M.; Kwoh, C.K.; Li, X.; Guan, C. Time-Series Representation Learning via Temporal and Contextual Contrasting. *arXiv* **2021**, arXiv:2106.14112. [CrossRef]
4. Woo, G.; Liu, C.; Sahoo, D.; Kumar, A.; Hoi, S. CoST: Contrastive Learning of Disentangled Seasonal-Trend Representations for Time Series Forecasting. *arXiv* **2022**, arXiv:2202.01575. [CrossRef]
5. Zheng, X.; Chen, X.; Schürch, M.; Mollaysa, A.; Allam, A.; Krauthammer, M. SimTS: Rethinking Contrastive Representation Learning for Time Series Forecasting. *arXiv* **2023**, arXiv:2303.18205. [CrossRef]
6. Bromley, J.; Guyon, I.; LeCun, Y.; Säckinger, E.; Shah, R. Signature verification using a "Siamese" time delay neural network. In *Proceedings of the 6th International Conference on Neural Information Processing Systems*; Morgan Kaufmann Publishers Inc.: San Francisco, CA, USA, 1993; NIPS'93, pp. 737–744.
7. Chen, X.; He, K. Exploring Simple Siamese Representation Learning. *arXiv* **2020**, arXiv:2011.10566. [CrossRef]
8. Franceschi, J.Y.; Dieuleveut, A.; Jaggi, M. Unsupervised Scalable Representation Learning for Multivariate Time Series. *arXiv* **2020**, arXiv:1901.10738. [CrossRef]
9. Tonekaboni, S.; Eytan, D.; Goldenberg, A. Unsupervised Representation Learning for Time Series with Temporal Neighborhood Coding. *arXiv* **2021**, arXiv:2106.00750. [CrossRef]
10. Luo, D.; Cheng, W.; Wang, Y.; Xu, D.; Ni, J.; Yu, W.; Zhang, X.; Liu, Y.; Chen, Y.; Chen, H.; et al. Time Series Contrastive Learning with Information-Aware Augmentations. *arXiv* **2023**, arXiv:2303.11911. [CrossRef]
11. Cleveland, R.B.; Cleveland, W.S.; McRae, J.E.; Terpenning, I. STL: A Seasonal-Trend Decomposition Procedure Based on Loess (with Discussion). *J. Off. Stat.* **1990**, *6*, 3–73.
12. Hyndman, R.; Athanasopoulos, G. *Forecasting: Principles and Practice*, 3rd ed.; OTexts: Melbourne, Australia, 2021.
13. Chernick, M.R. Wavelet Methods for Time Series Analysis. *Technometrics* **2001**, *43*, 491. [CrossRef]
14. Jolliffe, I.T.; Cadima, J. Principal component analysis: A review and recent developments. *Philos. Trans. R. Soc. A Math. Phys. Eng. Sci.* **2016**, *374*, 20150202. [CrossRef] [PubMed]

15. Yuan, L.; Wang, X.; Sun, Y.; Liu, X.; Dong, Z.Y. Multistep photovoltaic power forecasting based on multi-timescale fluctuation aggregation attention mechanism and contrastive learning. *Int. J. Electr. Power Energy Syst.* **2025**, *164*, 110389. [CrossRef]
16. Wang, M.; Meng, Y.; Sun, L.; Zhang, T. Decomposition combining averaging seasonal-trend with singular spectrum analysis and a marine predator algorithm embedding Adam for time series forecasting with strong volatility. *Expert Syst. Appl.* **2025**, *274*, 126864. [CrossRef]
17. Hong, J.; Han, P.; Rasool, A.; Chen, H.; Hong, Z.; Tan, Z.; Lin, F.; Wei, S.X.; Jiang, Q. A Correlational Strategy for the Prediction of High-Dimensional Stock Data by Neural Networks and Technical Indicators. In *Proceedings of the Big Data and Security*; Tian, Y., Ma, T., Jiang, Q., Liu, Q., Khan, M.K., Eds.; Springer: Singapore, 2023; pp. 405–419.
18. Chopra, S.; Hadsell, R.; LeCun, Y. Learning a similarity metric discriminatively, with application to face verification. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*; IEEE: New York, NY, USA, 2005; Volume 1, pp. 539–546. [CrossRef]
19. Grill, J.B.; Strub, F.; Altché, F.; Tallec, C.; Richemond, P.H.; Buchatskaya, E.; Doersch, C.; Pires, B.A.; Guo, Z.D.; Azar, M.G.; et al. Bootstrap your own latent: A new approach to self-supervised Learning. *arXiv* **2020**, arXiv:2006.07733. [CrossRef]
20. Zeng, A.; Chen, M.; Zhang, L.; Xu, Q. Are transformers effective for time series forecasting? In *Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence and Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence and Thirteenth Symposium on Educational Advances in Artificial Intelligence*; AAAI Press: Menlo Park, CA, USA, 2023; AAAI'23/IAAI'23/EAAI'23. [CrossRef]
21. Zhou, H.; Zhang, S.; Peng, J.; Zhang, S.; Li, J.; Xiong, H.; Zhang, W. Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting. *arXiv* **2021**, arXiv:2012.07436. [CrossRef]
22. Bai, S.; Kolter, J.Z.; Koltun, V. An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling. *arXiv* **2018**, arXiv:1803.01271. [CrossRef]
23. Chen, T.; Kornblith, S.; Norouzi, M.; Hinton, G. A Simple Framework for Contrastive Learning of Visual Representations. *arXiv* **2020**, arXiv:2002.05709. [CrossRef]
24. van den Oord, A.; Li, Y.; Vinyals, O. Representation Learning with Contrastive Predictive Coding. *arXiv* **2019**, arXiv:1807.03748. [CrossRef]


**Disclaimer/Publisher's Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.