C3RL: Rethinking the Combination of Channel-independence and Channel-mixing from Representation Learning

Shusen Ma[1], **Yunbo Zhao**[1,2,3*], **and Yu Kang**[1,2,3]

[1]Institute of Advanced Technology, University of Science and Technology of China, Shushan District, Hefei, Anhui, China
[2]Department of Automation, University of Science and Technology of China, Shushan District, Hefei, Anhui, China
[3]Institute of Artificial Intelligence, Hefei Comprehensive National Science Center, Shushan District, Hefei, Anhui, China
mss0913@mail.ustc.edu.cn, ybzhao@ustc.edu.cn

Abstract

Multivariate time series forecasting has drawn increasing attention due to its practical importance. Existing approaches typically adopt either channel-mixing (CM) or channel-independence (CI) strategies. CM strategy can capture inter-variable dependencies but fails to discern variable-specific temporal patterns. CI strategy improves this aspect but fails to fully exploit cross-variable dependencies like CM. Hybrid strategies based on feature fusion offer limited generalization and interpretability. To address these issues, we propose C3RL, a novel representation learning framework that jointly models both CM and CI strategies. Motivated by contrastive learning in computer vision, C3RL treats the inputs of the two strategies as transposed views and builds a siamese network architecture: one strategy serves as the backbone, while the other complements it. By jointly optimizing contrastive and prediction losses with adaptive weighting, C3RL balances representation and forecasting performance. Extensive experiments on seven models show that C3RL boosts the best-case performance rate to 81.4% for models based on CI strategy and to 76.3% for models based on CM strategy, demonstrating strong generalization and effectiveness.

Code — https://github.com/SSMa913/NICLab-C3RL
Extended version — https://arxiv.org/abs/2507.17454

Introduction

Multivariate time series forecasting (MTSF) has a wide range of real-world applications, including electricity consumption prediction (Qureshi, Arbab, and Rehman 2024), traffic flow forecasting (Kong, Guo, and Liu 2024), and vital signal prediction (Wang et al. 2024). Multivariate time series (MTS) data are rich in features, capturing both the temporal dynamics of individual variables and the inter-variable correlations. These features encode not only evolving trends but also intricate entanglements among variables. Effectively extracting and modeling these representations is critical to improving forecasting accuracy and remains a central challenge in MTSF research.

Currently, two dominant input processing strategies are widely adopted: *channel-mixing* (CM) and *channel-independence* (CI) strategies. CM treats multivariate observations at each time step as a single token, focusing on temporal dependencies. In contrast, CI approaches treat each variable's sequence as an independent token. Within this category, we identify two subtypes: *explicit channel independence* (ECI), where variables are sequentially fed into a shared backbone to learn temporal patterns of diverse variables (Nie et al. 2023), and *implicit channel independence* (ICI), where the sequence of shape (L, N) is transposed to (N, L) before being input as a whole (Ma et al. 2024b; Liu et al. 2024).

Recent studies suggest that CI models often outperform CM ones in terms of prediction accuracy (Liu et al. 2024; Zeng et al. 2023). However, relying solely on one strategy may fail to capture the full spectrum of temporal dependencies, especially for heterogeneous datasets (Ahamed and Cheng 2024). As a result, recent methods attempt to combine both strategies (Ahamed and Cheng 2024; Liang et al. 2024), primarily through feature fusion (Fan et al. 2025). Nonetheless, these fusion-based approaches often learn a single-task mapping focused on label prediction rather than robust representation learning. This leads to limited generalization, reduced interpretability, and deteriorated performance on unseen patterns.

Contrastive learning, as a form of self-supervised learning, aims to learn meaningful feature representations by modeling the similarity and dissimilarity between data samples. Extensive studies (Yue et al. 2022; Woo et al. 2022) have shown its effectiveness in learning useful temporal representations for time series forecasting. However, these studies predominantly rely on a single data processing strategy, especially the CM strategy. Therefore, integrating contrastive learning with both processing strategies is a promising direction. Such integration not only enhances the model's adaptability to diverse data types and scenarios but also improves its representational capacity. To achieve this, several key challenges must be addressed:

1) How to Reduce Training Cost while Avoiding Collapsing Solutions? A common issue in contrastive learning is representation collapse, where the model maps all inputs to identical or near-identical embeddings due to insufficient constraints on representational diversity (Chen and He 2021). SimCLR (Chen et al. 2020) mitigates this via negative sample pairs, but at the cost of increased computation.

*Corresponding author.

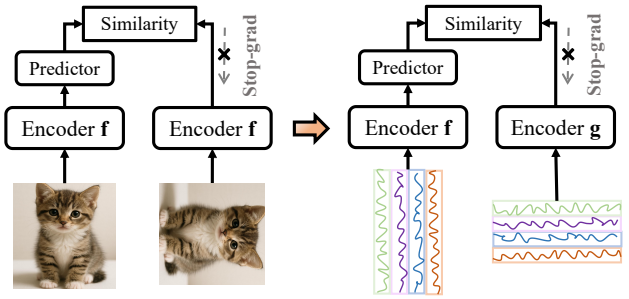

Figure 1: The application of the simple siamese networks (SimSiam) from the images (left) to the time series (right).

BYOL (Grill et al. 2020) uses only positive pairs, yet avoids collapse due to the use of a momentum encoder.

2) How to Construct Meaningful Positive Sample Pairs? In computer vision, positive sample pairs are typically created by applying data augmentations (e.g., rotation, cropping) to the same image (He et al. 2020), while samples from different images are treated as negatives (Chen et al. 2020). However, in the context of time series forecasting, how to effectively construct positive pairs remains underexplored, especially when jointly considering CM and CI strategies. There is a lack of systematic investigation into leveraging these two complementary views to define semantically meaningful sample pairs for contrastive learning.

3) How to Design an Effective Loss Function? Contrastive loss functions guide the model to pull together positive pairs and push apart negative ones, thereby learning discriminative features. However, since our ultimate goal is forecasting, the prediction loss—which measures the discrepancy between predicted and ground truth values—must also be incorporated. Balancing representation learning (via contrastive loss) and prediction accuracy (via forecasting loss) is thus a central challenge in loss design.

To address the aforementioned limitations, we rethink the **C**ombination of **C**hannel-independence and **C**hannel-mixing from a **R**epresentation **L**earning perspective, and propose a unified framework, **C3RL**.

1) SimSiam-inspired Architecture Design. To avoid collapsing solutions without increasing training cost by adding more data, C3RL adopts the SimSiam architecture (Chen and He 2021) as its design inspiration. As shown in the left part of Figure 1, SimSiam effectively avoids representational collapse by applying a stop-gradient on one branch, eliminating the need for negative samples. In SimSiam, both inputs share the same encoder. However, in C3RL, since the time series inputs differ in dimensional structure due to transposition, the same encoder cannot process both views consistently. To solve this, we introduce a siamese encoder g corresponding to the backbone encoder f, as illustrated on the right side of Figure 1. The siamese encoder is designed to preserve the architecture of f, with internal feature dimensions adjusted only to accommodate the input shape.

2) Channel Views as Positive Sample Pairs. Upon analysis, we observe that the inputs under the CM and ICI strategies can be viewed as transposed versions of each other. This rotation-like transformation aligns with the semantics of positive pairs in SimSiam, and thus, they can be naturally treated as positive examples within our contrastive learning setup (see Figure 1, right). The application of C3RL to the ECI can see the **Appendix**.

3) Joint Training for Representation and Forecasting. To enhance representation learning while maintaining strong performance on downstream forecasting tasks, C3RL adopts a joint optimization strategy, where SimSiam-based contrastive learning and supervised prediction are trained simultaneously. Given that the importance of contrastive and predictive signals can vary across datasets and tasks, we introduce adaptive weights to balance the contrastive loss and the prediction loss dynamically. This enables the model to adaptively trade off between representational richness and task-specific accuracy, leading to superior generalization across diverse scenarios.

The main contributions of this paper are summarized as follows:

- We propose a novel paradigm, C3RL, based on representation learning, which unifies CM and CI strategies within a new framework. This approach significantly improves the performance and enhances the representation ability of most mainstream forecasting models and offers a new modeling perspective for future research.
- We design a new loss function that integrates contrastive loss and forecasting loss, enabling dynamic adjustment of their relative weights. This allows the model to maintain an optimal balance between representation quality and forecasting accuracy across different tasks.
- We validate the effectiveness of C3RL on nine publicly available real-world datasets using five CI models and two CM models. Specifically, C3RL boosts the best-case performance rate from 43.6% to 81.4% for CI models, and from 23.8% to 76.3% for CM models, demonstrating strong generalization and effectiveness.

Related Work

Traditional time series forecasting models, such as ARIMA (Zhang 2003) and SVM (Cao 2003), are simple and effective in certain scenarios. However, they often struggle to capture the complex nonlinear dependencies among multiple variables. With the increasing availability of data and the rapid development of deep learning techniques, various neural network architectures, such as Recurrent Neural Networks (Hewamalage, Bergmeir, and Bandara 2021) and Convolutional Neural Networks (Ma et al. 2023), have been increasingly applied to time series forecasting tasks. In particular, the emergence of the Transformer (Vaswani et al. 2017) has sparked extensive research efforts aimed at leveraging its strength in modeling long-term dependencies to improve forecasting performance. Informer (Zhou et al. 2021) stands out as a milestone in MTSF, adopting a typical CM strategy and introducing a generative-style decoder that enables long-sequence prediction in a single forward pass. Building on this paradigm, numerous models have been proposed, including Autoformer (Wu et al. 2021),

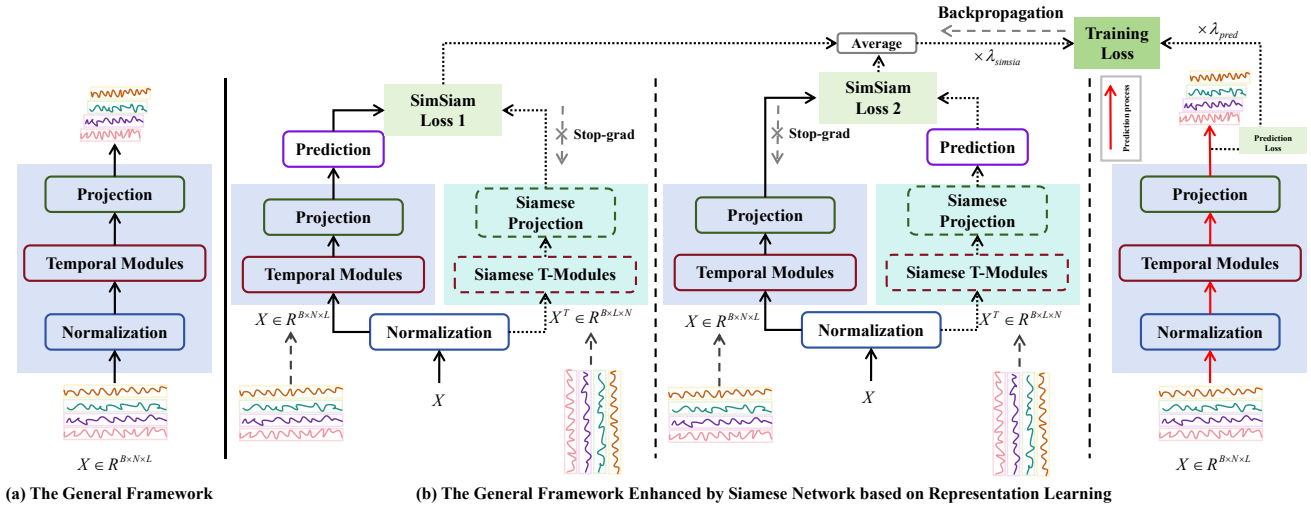

Figure 2: The pipeline of the combination of CI and CM by representation learning (C3RL). The T-Modules denotes the temporal modules

FEDformer (Zhou et al. 2022), and Pyraformer (Liu et al. 2022). However, the CM strategy, which treats the multivariate information at each time step as a single token, often leads to excessive coupling between variables and undermines predictive accuracy. To address this issue, PCDformer (Ma et al. 2024b) introduces a parallel convolutional mechanism to independently model each variable, reducing inter-variable interference. iTransformer (Liu et al. 2024) employs an inversion of input data and relies solely on the Transformer encoder, achieving competitive performance. In parallel, lightweight models based on multi-layer perceptrons (MLPs), such as DLinear (Zeng et al. 2023) and RLinear (Li et al. 2023), have demonstrated the ability to outperform some Transformer-based approaches. More recently, the advent of Mamba, with its State Space Model (SSM) structure and parallel scan mechanism, has shown promising results in both computational efficiency and forecasting accuracy (Ma et al. 2024a). While these methods improve upon the traditional CM strategy by better highlighting variable-specific features, they still differ from the strategy adopted in PatchTST (Nie et al. 2023), where each variable is modeled in a completely independent manner, referred to as the ECI strategy. In contrast, we term the aforementioned approaches as adopting an ICI strategy.

Although CI modeling generally outperforms CM-based models, relying solely on a single strategy is insufficient to fully capture the complex patterns embedded in MTS data (Ahamed and Cheng 2024). As a result, recent studies (Ahamed and Cheng 2024; Liang et al. 2024; Fan et al. 2025) have explored hybrid approaches that combine CM and CI strategies to build more generalizable forecasting frameworks. However, most of these works focus primarily on improving predictive accuracy through strategy fusion, often overlooking the model's representation capacity and interpretability. Therefore, it remains a promising direction to investigate how to effectively integrate CM and CI strategies into a unified framework that not only enhances forecasting performance but also improves the model's representational power and interpretability.

Methodology

In this work, we rethink and propose the C3RL, a novel paradigm for model enhancement that can be seamlessly applied to mainstream MTSF models. The overall pipeline of C3RL is illustrated in Figure 2. As shown in Figure 2(a), a general time series forecasting model typically consists of a normalization layer (e.g., Layer normalization (Ba, Kiros, and Hinton 2016) or RevIN (Kim et al. 2021)), temporal modules (such as linear-based, Mamba-based, or Transformer-based architectures), and a projection layer (i.e., prediction head). As introduced in Introduction, we extend this baseline framework by designing a siamese-network architecture, where each branch adopts a different channel strategy. Furthermore, we introduce a joint loss function based on weight to balance the model's representational and predictive capabilities. The detailed pipeline of our proposed framework is shown in Figure 2(b).

Preliminary

Let $X = [x_1, x_2, \ldots, x_L] \in \mathbb{R}^{L\times N}$ denote an MTS data, where L is the total number of time steps and N represents the number of variables (or channels). Each observation at time step t is denoted by $x_t \in \mathbb{R}^N$. The MTSF task aims to learn a mapping function f that predicts the future sequence $Y = [\hat{x}_{L+1}, \ldots, \hat{x}_{L+P}] \in \mathbb{R}^{P\times N}$, where P is the forecasting horizon. Specifically, models that adopt the CM strategy take inputs in the form of $X = [x_1, x_2, \ldots, x_L] \in \mathbb{R}^{L\times N}$. In contrast, models that use an ICI strategy receive inputs as $X = [x_1, x_2, \ldots, x_N] \in \mathbb{R}^{N\times L}$. For models with an ECI strategy, the multivariate input needs to be completely separated beforehand, transforming the input from

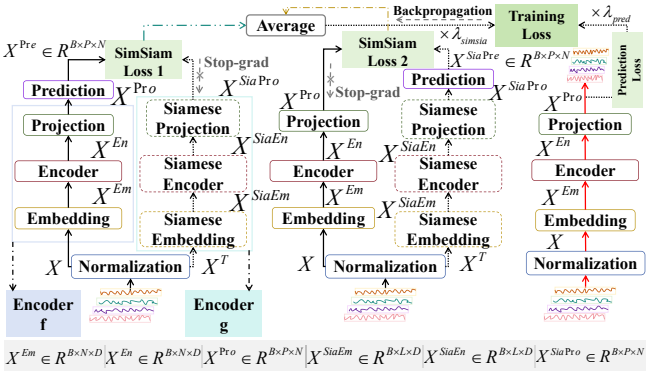

Figure 3: The application of C3RL to the iTransformer.

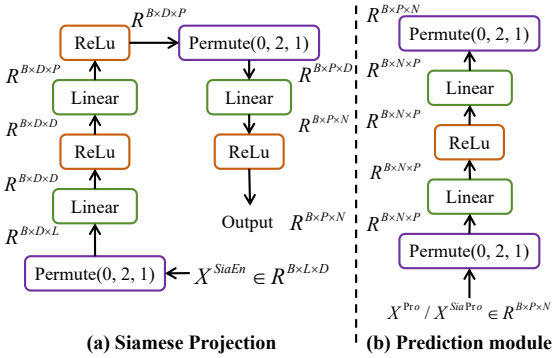

Figure 4: The structure of the Siamese Projection (left) and Prediction module (right).

$X \in \mathbb{R}^{B\times N\times L}$ to $X \in \mathbb{R}^{(B\cdot N)\times 1\times L}$, where B is the batch size.

Siamese Network

The core idea of C3RL is to enhance the representational and predictive capabilities of existing backbone models by constructing a corresponding Siamese network, as shown in Figure 2(b). This Siamese network comprises two components: Siamese Temporal Modules (T-Modules) and a Siamese Projection. The T-Modules are designed by replicating the temporal structure of the backbone while adjusting only the internal feature dimensions to meet input requirements, avoiding the need for newly engineered feature extractors. The Siamese Projection is implemented with a minimal MLP, aiming to align its output features with those of the backbone's projection head using the fewest possible layers. For clarity, Figure 3 presents the construction of a Siamese network by incorporating C3RL into iTransformer (more examples shown in the **Appendix**):

Step 1: In iTransformer, the Embedding and Encoder layers are considered as part of the T-Modules. These components can be directly transferred to construct the Siamese Embedding and Siamese Encoder layers. *Step 2:* Since the input dimensions on the Siamese side differ, the internal parameters of the Siamese Embedding layer must be adjusted accordingly to ensure proper data flow. Specifically, the input feature dimension of the original Embedding layer (implemented via a linear layer) is L, so the linear layer is defined as $nn.Linear(D_{in}, D)$ with $D_{in} = L$. In contrast, the input feature dimension for the Siamese Embedding layer is N, and thus D_{in} should be set to N in the corresponding linear layer. *Step 3:* The output of the Siamese Embedding layer, which serves as the input to the Siamese Encoder, has the same last-dimensional size D as the input to the original Encoder. Therefore, the internal structure of the Siamese Encoder remains unchanged. *Step 4:* The output of the original Projection layer is denoted as $X^{Pro} \in \mathbb{R}^{B\times P\times N}$. To align the output of the Siamese Encoder, $X^{SiaEn} \in \mathbb{R}^{B\times L\times D}$, with X^{Pro} in feature space, we introduce a Siamese Projection layer, as illustrated in Figure 4(a).

Training and Prediction

In this section, we introduce the formulation of the training loss and the prediction process of the model. As illustrated in Figure 3, X and X^T are used as the inputs to the Encoder f and g, respectively. We define the output of the Encoder f and Encoder g as $X^{Pro} \triangleq Enc_f(X)$ and $X^{SiaPro} \triangleq Enc_g(X^T)$, respectively. Prediction module is used to transform the output of one of the Encoder and match it to the view of the other Encoder. The structure of the Prediction module is shown as Figure 4(b). As shown on the left side of Figure 3, we define $X^{Pre} \triangleq P(X^{Pro})$, where $P(\cdot)$ denotes the Prediction module. Following (Chen and He 2021), we minimize the negative cosine similarity of the X^{Pre} and X^{SiaPro}:

$$\mathcal{D}(X^{Pre}, X^{SiaPro}) = -\frac{X^{Pre}}{\|X^{Pre}\|_2} \cdot \frac{X^{SiaPro}}{\|X^{SiaPro}\|_2}, \quad (1)$$

where $\|\cdot\|_2$ denotes the l_2-norm. Following (Grill et al. 2020), we define a symmetrized loss as:

$$\mathcal{L}_{simsia} = \frac{1}{2}\cdot\mathcal{D}(X^{Pre}, X^{SiaPro}) + \frac{1}{2}\cdot\mathcal{D}(X^{Pro}, X^{SiaPre}), \quad (2)$$

where X^{SiaPre} represents the matching of the Encoder g to the Encoder f by the Prediction module shown in the middle of Figure 3. Following (Chen and He 2021), we adopt the stop-gradient (stop-grad) operation to avoid the collapsing solutions. Therefore, we need to modify the equation (1) to:

$$\begin{aligned}&\mathcal{D}(X^{Pre}, \text{stop-grad}(X^{SiaPro}))\\ &= -\frac{X^{Pre}}{\|X^{Pre}\|_2} \cdot \text{stop-grad}\left(\frac{X^{SiaPro}}{\|X^{SiaPro}\|_2}\right),\end{aligned} \quad (3)$$

which means that X^{SiaPro} is excluded from gradient computation. Similarly, we redefine the equation (2) as:

$$\begin{aligned}\mathcal{L}_{simsia} &= \frac{1}{2}\cdot\mathcal{D}(X^{Pre}, \text{stop-grad}(X^{SiaPro}))\\ &+ \frac{1}{2}\cdot\mathcal{D}(\text{stop-grad}(X^{Pro}), X^{SiaPre}).\end{aligned} \quad (4)$$

$\mathcal{L}_{simsia}$ is a contrastive loss function designed to pull together features extracted using different data augmentation

Model		S-Mamba		**+ C3RL**		DLinear		**+ C3RL**		PatchTST		**+ C3RL**		iTransformer		**+ C3RL**		RLinear		**+ C3RL**	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	0.388	0.406	**0.386**	**0.405**	0.384	0.405	**0.374**	**0.395**	**0.375**	**0.399**	0.376	0.400	**0.387**	**0.405**	**0.387**	**0.405**	**0.366**	**0.391**	0.367	**0.391**
	192	0.445	**0.441**	**0.442**	0.442	0.443	0.450	**0.408**	**0.418**	**0.414**	**0.421**	**0.414**	**0.421**	**0.441**	**0.436**	**0.441**	0.438	**0.403**	**0.412**	0.404	**0.412**
	336	0.490	**0.465**	**0.485**	**0.465**	0.447	0.448	**0.443**	**0.444**	0.431	0.435	**0.429**	**0.434**	0.491	0.462	**0.485**	**0.461**	0.420	**0.423**	**0.418**	**0.423**
	720	0.506	0.497	**0.501**	**0.494**	0.504	0.515	**0.472**	**0.489**	0.450	0.466	**0.445**	**0.463**	0.509	0.494	**0.503**	**0.491**	0.442	0.456	**0.431**	**0.449**
ETTh2	96	0.297	0.349	**0.295**	**0.347**	0.290	0.353	**0.280**	**0.349**	**0.274**	0.336	**0.274**	**0.335**	0.301	0.350	**0.297**	**0.349**	0.262	0.331	**0.260**	**0.328**
	192	0.378	0.399	**0.376**	**0.398**	0.388	0.422	**0.362**	**0.403**	**0.338**	**0.378**	**0.338**	0.379	0.380	0.399	**0.378**	**0.398**	0.320	0.374	**0.317**	**0.369**
	336	0.425	0.435	**0.423**	**0.434**	0.463	0.473	**0.436**	**0.455**	0.331	0.380	**0.327**	**0.378**	0.424	**0.432**	**0.422**	**0.432**	0.326	0.388	**0.324**	**0.385**
	720	0.432	0.448	**0.431**	**0.447**	0.733	0.606	**0.667**	**0.580**	0.379	0.421	**0.375**	**0.418**	0.430	**0.447**	**0.429**	**0.447**	0.425	0.449	**0.412**	**0.443**
ETTm1	96	**0.331**	**0.368**	0.332	0.369	0.301	0.345	**0.300**	**0.344**	0.292	0.343	**0.289**	**0.342**	**0.342**	0.377	**0.342**	**0.376**	**0.301**	**0.343**	0.302	**0.343**
	192	0.378	**0.393**	**0.377**	**0.393**	**0.336**	**0.366**	0.337	0.367	**0.331**	**0.369**	0.332	**0.369**	0.383	0.396	**0.381**	**0.394**	0.341	0.367	**0.338**	**0.365**
	336	0.410	**0.414**	**0.408**	**0.414**	**0.372**	**0.389**	0.375	0.391	**0.365**	0.392	0.366	**0.391**	0.418	0.418	**0.416**	**0.417**	0.374	0.386	**0.371**	**0.384**
	720	**0.474**	**0.451**	0.475	0.453	**0.427**	**0.423**	0.432	0.427	0.421	0.425	**0.420**	**0.424**	0.487	0.457	**0.484**	**0.454**	0.430	0.418	**0.426**	**0.415**
ETTm2	96	0.182	0.266	**0.179**	**0.264**	0.172	**0.267**	**0.171**	0.268	0.164	0.254	**0.163**	**0.252**	0.186	0.272	**0.183**	**0.267**	**0.164**	0.253	**0.164**	**0.252**
	192	0.252	0.313	**0.248**	**0.309**	0.237	**0.314**	**0.235**	0.317	0.221	**0.292**	**0.220**	**0.292**	0.254	0.314	**0.251**	**0.312**	**0.219**	**0.290**	**0.219**	**0.290**
	336	0.313	0.349	**0.311**	**0.348**	**0.295**	**0.359**	0.313	0.373	0.278	**0.329**	**0.277**	0.330	0.316	**0.351**	**0.315**	**0.351**	**0.273**	**0.326**	0.274	**0.326**
	720	**0.413**	**0.405**	**0.413**	**0.405**	0.427	0.439	**0.392**	**0.411**	0.367	0.385	**0.362**	**0.382**	0.414	0.407	**0.410**	**0.405**	**0.366**	0.385	0.367	**0.384**
Exchange	96	**0.086**	**0.206**	0.088	0.208	0.085	0.209	**0.082**	**0.202**	**0.093**	**0.213**	**0.093**	**0.213**	**0.086**	**0.206**	0.087	0.208	**0.090**	**0.209**	**0.090**	**0.209**
	192	0.182	0.304	**0.180**	**0.302**	0.162	0.296	**0.160**	**0.291**	**0.194**	**0.314**	**0.194**	**0.314**	0.181	0.304	**0.178**	**0.302**	0.193	0.311	**0.186**	**0.305**
	336	**0.331**	**0.417**	**0.331**	**0.417**	0.333	0.441	**0.319**	**0.435**	0.355	0.436	**0.352**	**0.433**	**0.338**	**0.422**	**0.338**	**0.422**	0.362	0.435	**0.361**	**0.434**
	720	**0.858**	**0.699**	0.862	0.701	0.898	0.725	**0.888**	**0.721**	**0.903**	**0.712**	0.904	0.713	0.853	**0.696**	**0.852**	0.697	0.923	**0.719**	**0.919**	**0.719**
Illness	24	1.849	0.865	**1.808**	**0.854**	**2.280**	**1.061**	2.289	1.064	**1.401**	**0.738**	1.522	0.815	2.358	1.063	**2.345**	**1.051**	2.306	1.037	**2.303**	**1.032**
	36	2.188	1.006	**1.776**	**0.877**	**2.235**	**1.059**	2.274	1.063	1.452	0.839	**1.317**	**0.781**	2.184	1.002	**2.182**	**0.999**	2.173	1.014	**2.069**	**0.977**
	48	**1.708**	**0.875**	1.787	0.894	2.298	**1.079**	**2.285**	1.087	1.679	0.859	**1.498**	**0.817**	**2.124**	1.019	2.161	**1.018**	2.184	1.005	**2.151**	**0.996**
	60	2.338	1.005	**2.068**	**0.953**	2.573	1.157	**2.420**	**1.111**	1.621	0.888	**1.395**	**0.798**	2.165	1.032	**2.061**	**0.991**	2.087	0.983	**2.080**	**0.981**

Table 1: Results on base models based on CI. The better performance in each setting is shown in **bold**.

strategies (ICI and ECI) in the feature space. This encourages the model to learn richer temporal patterns and enhances its representational capacity. However, since the ultimate objective is forecasting, the construction of the training loss must also incorporate a prediction error term. To balance the influence of contrastive and error-based losses, we assign them different weights and dynamically adjust their contributions in the training. This strategy aims to enhance the model's representation ability while improving the model's final predictive performance. The overall training loss is defined as follows:

$$\mathcal{L}_{\text{total}} = \lambda_{simsia} \cdot \mathcal{L}_{simsia} + \lambda_{pred} \cdot \mathcal{L}_{pred}, \quad (5)$$

where λ_{simsia} and λ_{pred} are tunable hyperparameters. The inference (prediction) process of the trained model, indicated by the red arrow on the right side of Figure 3, remains consistent with that of iTransformer.

Experiments

Experimental Settings

Datasets. We conduct extensive experiments on nine publicly available and widely used datasets: ETT (ETTh1, ETTh2, ETTm1 and ETTm2) (Zhou et al. 2021), Exchange (Lai et al. 2018), Weather, Electricity, Traffic, and Illness (Wu et al. 2021).

Evaluation metrics. We adopt two widely used evaluation metrics from existing benchmarking protocols: Mean Squared Error (MSE) and Mean Absolute Error (MAE), to compare the performance of mainstream models with their enhanced versions integrated with C3RL.

Compared baselines. Since models based on the CI strategy have achieved state-of-the-art performance, most recent studies have been built upon this approach (Ma et al. 2024a; Liang et al. 2024; Fan et al. 2025). Therefore, our main comparisons and analyses focus on models following this strategy. Specifically, we select three representative models constructed under different neural network paradigms: MLP-based (DLinear (Zeng et al. 2023) and RLinear (Li et al. 2023)), Transformer-based (iTransformer (Liu et al. 2024) and PatchTST (Nie et al. 2023)), and Mamba-based (S-Mamba (Wang et al. 2025)). In addition, to evaluate the generalization capability of C3RL, we include two representative models based on the CM strategy: Informer (Zhou et al. 2021) and Autoformer (Wu et al. 2021).

Experimental details. All experiments are implemented in PyTorch and conducted on a single NVIDIA GeForce RTX 3090 GPU. To ensure fair comparisons, we reimplement all baselines and their C3RL-enhanced variants under the same hardware and software settings. The hyperparameters for baselines follow their official implementations. For the baseline implementations, we kept the original random seed values unchanged. For those without a predefined seed, we uniformly set the random seed to 2025.

Model		Informer		+ C3RL		Autoformer		+ C3RL	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	24(24)	0.567	0.550	**0.486**	**0.500**	0.381	**0.422**	**0.376**	0.424
	48(48)	0.674	**0.613**	**0.657**	0.617	0.391	0.419	**0.381**	**0.413**
	168(168)	1.049	0.831	**1.000**	**0.777**	0.502	0.483	**0.458**	**0.457**
	336(336)	1.434	0.990	**1.152**	**0.865**	0.511	0.488	**0.492**	**0.480**
	720(720)	1.388	0.942	**1.450**	**0.940**	**0.499**	**0.501**	0.500	0.504
ETTh2	24(24)	0.761	0.669	**0.408**	**0.503**	**0.259**	**0.339**	0.278	0.360
	48(48)	**1.863**	**1.096**	1.923	1.097	0.314	**0.374**	**0.312**	0.376
	168(168)	4.152	1.727	**2.726**	**1.391**	0.491	0.471	**0.427**	**0.432**
	336(336)	3.736	1.605	**2.942**	**1.433**	0.483	0.486	**0.459**	**0.468**
	720(720)	3.821	1.648	**3.070**	**1.509**	**0.478**	**0.487**	0.483	0.500
ETTm1	24(24)	0.408	0.421	**0.325**	**0.376**	**0.396**	**0.425**	0.409	0.426
	48(48)	0.444	0.469	**0.416**	**0.421**	0.476	0.455	**0.467**	**0.454**
	96(96)	0.520	0.528	**0.508**	**0.509**	0.467	**0.452**	**0.443**	0.455
	288(288)	0.907	0.746	**0.838**	**0.689**	0.584	0.519	**0.574**	**0.509**
	672(672)	1.030	0.822	**0.991**	**0.791**	0.552	**0.508**	**0.540**	0.510
Weather	24(48)	0.108	0.179	**0.107**	**0.173**	0.230	**0.311**	**0.230**	0.316
	48(96)	0.216	0.314	**0.191**	**0.286**	0.260	0.331	**0.246**	**0.318**
	168(192)	0.330	0.374	**0.294**	**0.356**	0.344	0.393	**0.295**	**0.354**
	336(336)	0.443	**0.453**	**0.414**	0.460	0.359	0.395	**0.355**	**0.392**
	720(720)	**0.526**	**0.497**	0.589	0.550	0.415	0.423	**0.413**	**0.421**
1st	count	2	4	**18**	**16**	4	9	**16**	**11**

Table 2: Results on base models based on CM. The forecasting horizon in parentheses is specific to Autoformer.

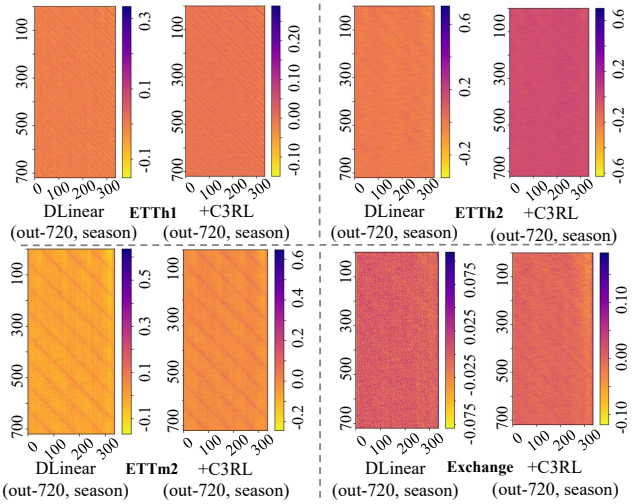

Figure 5: Visualization of the weights of DLinear on several datasets. Out-720 denotes that the prediction horizon is 720. Season means the seasonal item obtained by decomposition.

Results and Analysis

Quantitative analysis. We conduct a comprehensive evaluation of C3RL on nine datasets across seven base models and their C3RL-enhanced counterparts, using MSE and MAE as metrics. As shown in Table 1 (full results shown in the **Appendix**), the base models enhanced by C3RL consistently demonstrate improved predictive performance. Specifically, S-Mamba achieves the best performance in only 25 out of 72 metrics (12 for MSE and 13 for MAE), making a best-score rate of 34.7%, while S-Mamba (+C3RL) achieves 59 best scores (28 for MSE, 31 for MAE), raising the rate to 81.9%. DLinear obtains 31 best results (14 for MSE, 17 for MAE), with a rate of 43.1%, whereas DLinear (+C3RL) improves to 52 best results (28 for MSE, 24 for MAE), reaching 72.2%. PatchTST achieves 24 best scores out of 56 metrics (13 for MSE, 11 for MAE), with a best-score rate of 42.9%. In contrast, PatchTST (+C3RL) reaches 46 best scores (23 for MSE, 23 for MAE), achieving 82.1%. iTransformer obtains 30 best results (13 for MSE, 17 for MAE) with a 41.7% rate, while iTransformer (+C3RL) achieves 59 best scores (32 for MSE, 27 for MAE), pushing the rate to 81.9%. RLinear reaches 40 best scores (20 for MSE, 20 for MAE), with a 55.6% best-score rate. Enhanced with C3RL, it obtains 64 best scores (30 for MSE, 34 for MAE), increasing the rate to 88.9%. Overall, C3RL achieves an optimal rate of **81.4%** across all tasks on the five models, representing a **37.8%** improvement compared to the **43.6%** achieved by models without C3RL. In addition, the results in Table 2 show that applying C3RL to models based on CM strategies also yields strong performance (with the overall optimal rate increasing from **23.8%** to **76.3%**), and significant improvements are observed in many tasks. The selection of $\{\lambda_{simsia}, \lambda_{pred}\}$ is shown in the **Appendix**.

Qualitative analysis. Figure 5 illustrates the seasonal weight visualizations of DLinear and DLinear (+C3RL). It is evident that DLinear (+C3RL) exhibits stronger structural clarity and greater discriminability in its visual representations, indicating enhanced feature modeling capability. For instance, in the ETTh2 and ETTm2 datasets, the weight heatmaps of DLinear (+C3RL) display overall deeper color intensities, suggesting significantly increased attention to specific time steps. Notably, in the ETTh1 and Exchange datasets, DLinear (+C3RL) demonstrates more distinct, periodically arranged dark band patterns, directly validating the effectiveness of C3RL in capturing seasonal patterns and improving representational expressiveness. To further illustrate the impact of C3RL on prediction performance, we visualize the forecasting results of DLinear on two datasets in Figure 6 (more results shown in the **Appendix**). As shown, incorporating C3RL enables the predicted curves to align more closely with the ground truth, indicating improved fitting accuracy. This is particularly prominent in the ETTh1 (out-192), where C3RL substantially enhances the model's ability to capture complex temporal patterns, realizing more accurate forecasting of future trends. These results collectively validate the effectiveness of C3RL in improving the temporal representation capacity of the model.

Ablation Studies

We conduct a comprehensive evaluation of five base models integrated with C3RL under various hyperparameter combinations $\{\lambda_{simsia}, \lambda_{pred}\}$, with the prediction length set to 96. The experimental results are illustrated in Figure 7, where WEIGHT denotes λ_{simsia} and the constraint $\lambda_{simsia} + \lambda_{pred} = 1$ holds. The symbol "+" indicates base models equipped with C3RL. As shown in the fig-

Model		S-Mamba+*		**S-Mamba+**		DLinear+*		**DLinear+**		PatchTST+*		**PatchTST+**		iTrans+*		**iTrans+**		RLinear+*		**RLinear+**	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	0.402	0.420	**0.386**	**0.405**	**0.374**	**0.395**	**0.374**	**0.395**	0.379	0.403	**0.376**	**0.400**	0.396	0.414	**0.387**	**0.405**	0.383	0.406	**0.367**	**0.391**
	192	0.449	0.447	**0.442**	**0.442**	0.409	**0.418**	**0.408**	**0.418**	0.415	0.422	**0.414**	**0.421**	0.446	0.441	**0.441**	**0.438**	0.413	0.420	**0.404**	**0.412**
	336	0.490	0.468	**0.485**	**0.465**	0.465	0.464	**0.443**	**0.444**	**0.429**	0.435	**0.429**	**0.434**	0.491	0.465	**0.485**	**0.461**	0.428	0.435	**0.418**	**0.423**
	720	**0.501**	**0.494**	**0.501**	**0.494**	0.544	0.537	**0.472**	**0.489**	**0.444**	**0.462**	0.445	0.463	**0.503**	**0.491**	**0.503**	**0.491**	0.437	0.454	**0.431**	**0.449**
ETTh2	96	0.307	0.357	**0.295**	**0.347**	0.288	0.354	**0.280**	**0.349**	0.276	0.337	**0.274**	**0.335**	0.302	0.351	**0.297**	**0.349**	0.261	**0.328**	**0.260**	**0.328**
	192	0.386	0.404	**0.376**	**0.398**	0.367	0.407	**0.362**	**0.403**	0.339	**0.379**	**0.338**	**0.379**	0.380	0.399	**0.378**	**0.398**	0.319	0.370	**0.317**	**0.369**
	336	0.426	0.436	**0.423**	**0.434**	0.472	0.477	**0.436**	**0.455**	**0.326**	**0.378**	0.327	**0.378**	0.429	0.436	**0.422**	**0.432**	0.327	0.387	**0.324**	**0.385**
	720	0.435	0.451	**0.431**	**0.447**	0.685	0.587	**0.667**	**0.580**	**0.375**	**0.418**	**0.375**	**0.418**	0.433	0.450	**0.429**	**0.447**	0.424	0.449	**0.412**	**0.443**
ETTm2	96	0.186	0.271	**0.179**	**0.264**	0.172	**0.268**	**0.171**	**0.268**	0.170	0.258	**0.163**	**0.252**	0.184	0.270	**0.183**	**0.267**	0.165	0.254	**0.164**	**0.252**
	192	0.255	0.315	**0.248**	**0.309**	**0.235**	**0.317**	**0.235**	**0.317**	0.226	0.297	**0.220**	**0.292**	0.254	0.314	**0.251**	**0.312**	0.223	0.294	**0.219**	**0.290**
	336	0.317	0.353	**0.311**	**0.348**	0.318	0.379	**0.313**	**0.373**	0.281	0.333	**0.277**	**0.330**	0.317	0.353	**0.315**	**0.351**	0.278	0.330	**0.274**	**0.326**
	720	0.416	0.407	**0.413**	**0.405**	0.393	0.412	**0.392**	**0.411**	0.364	0.383	**0.362**	**0.382**	0.413	0.406	**0.410**	**0.405**	0.369	0.385	**0.367**	**0.384**
Exchange	96	0.092	0.215	**0.088**	**0.208**	0.085	0.208	**0.082**	**0.202**	0.099	0.222	**0.093**	**0.213**	0.092	0.215	**0.087**	**0.208**	0.095	0.219	**0.090**	**0.209**
	192	0.188	0.310	**0.180**	**0.302**	0.165	0.298	**0.160**	**0.291**	0.196	0.317	**0.194**	**0.314**	0.186	0.308	**0.178**	**0.302**	0.195	0.317	**0.186**	**0.305**
	336	0.343	0.426	**0.331**	**0.417**	0.365	0.456	**0.319**	**0.435**	0.356	0.435	**0.352**	**0.433**	0.352	0.432	**0.338**	**0.422**	**0.357**	0.435	0.361	**0.434**
	720	0.877	0.710	**0.862**	**0.701**	1.022	0.766	**0.888**	**0.721**	0.906	**0.713**	**0.904**	**0.713**	0.870	0.705	**0.852**	**0.697**	0.923	0.720	**0.919**	**0.719**

Table 3: Performance comparison between weighted and unweighted base models enhanced by C3RL. "+" means the base models with C3RL and "*" denotes the models without weighting. iTrans is the iTransformer.

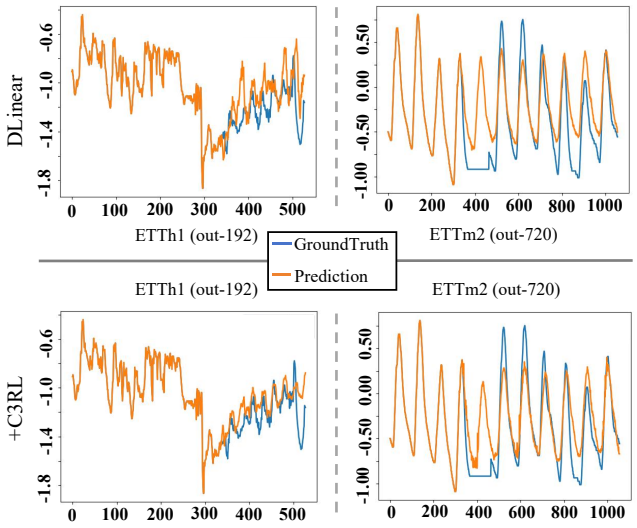

Figure 6: Visualization of the predictive performance.

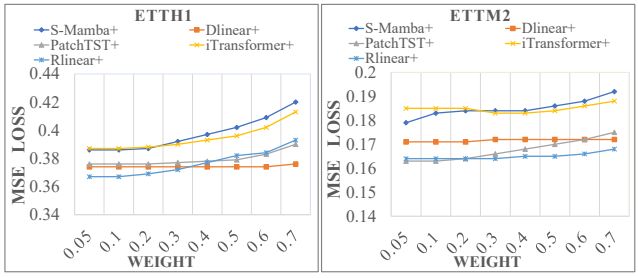

Figure 7: Influence of diverse WEIGHT (λ_{simsia}) on the performance of base models enhanced by C3RL.

ure, increasing λ_{simsia}—i.e., placing more emphasis on representation learning during training—leads to a gradual rise in prediction error. This observation suggests that over-optimizing the representational capacity may compromise the model's ability to perform downstream forecasting. Therefore, achieving a proper trade-off between representation learning and prediction objectives during joint training is crucial for enhancing overall model performance.

Furthermore, we set both $\lambda_{simsiam}$ and λ_{pred} to 1 to evaluate the effectiveness of the proposed weighting mechanism. As shown in Table 3, it is evident that in nearly all tasks, the model incorporating C3RL with the weighting mechanism achieves significantly lower prediction errors compared to the baseline model that uses C3RL without weighting. These results demonstrate that appropriately balancing the contrastive and predictive losses not only preserves the representational capacity but also enhances performance on downstream forecasting tasks.

Conclusion

This paper introduces C3RL, a representation-learning–based framework for MTSF that unifies CM- and CI-based modeling strategies. Inspired by the input requirements of the two strategies and the learning scheme of SimSiam, C3RL constructs universal Siamese structures with complementary channel processing to align feature representations across heterogeneous input formats while avoiding representation collapse. Extensive experiments verify that C3RL consistently boosts forecasting accuracy and enhances the model's ability to capture diverse structural patterns. Overall, C3RL provides a new and effective paradigm for optimizing MTSF models, with strong potential for practical deployment.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (No. 62173317) and Dreams Foundation of Jianghuai Advance Technology Center (No. 2023-ZM01G003).

References

Ahamed, M. A.; and Cheng, Q. 2024. Timemachine: A time series is worth 4 mambas for long-term forecasting. In *ECAI 2024*, 1688–1695. IOS Press.

Ba, J. L.; Kiros, J. R.; and Hinton, G. E. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Cao, L. 2003. Support vector machines experts for time series forecasting. *Neurocomputing*, 51: 321–339.

Chen, T.; Kornblith, S.; Norouzi, M.; and Hinton, G. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, 1597–1607.

Chen, X.; and He, K. 2021. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 15750–15758.

Fan, B.; Ma, S.; Zhao, Y.-B.; and Kang, Y. 2025. DC-Mamber: A Dual Channel Prediction Model based on Mamba and Linear Transformer for Multivariate Time Series Forecasting. *arXiv preprint arXiv:2507.04381*.

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. 2020. Bootstrap your own latent a new approach to self-supervised learning. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, volume 33, 21271–21284.

He, K.; Fan, H.; Wu, Y.; Xie, S.; and Girshick, R. 2020. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 9729–9738.

Hewamalage, H.; Bergmeir, C.; and Bandara, K. 2021. Recurrent neural networks for time series forecasting: Current status and future directions. *International Journal of Forecasting*, 37(1): 388–427.

Kim, T.; Kim, J.; Tae, Y.; Park, C.; Choi, J.-H.; and Choo, J. 2021. Reversible instance normalization for accurate time-series forecasting against distribution shift. In *International conference on learning representations*.

Kong, W.; Guo, Z.; and Liu, Y. 2024. Spatio-temporal pivotal graph neural networks for traffic flow forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, 8627–8635.

Lai, G.; Chang, W.-C.; Yang, Y.; and Liu, H. 2018. Modeling long-and short-term temporal patterns with deep neural networks. In *The 41st international ACM SIGIR conference on research & development in information retrieval*, 95–104.

Li, Z.; Qi, S.; Li, Y.; and Xu, Z. 2023. Revisiting long-term time series forecasting: An investigation on linear mapping. *arXiv preprint arXiv:2305.10721*.

Liang, A.; Jiang, X.; Sun, Y.; Shi, X.; and Li, K. 2024. Bi-mamba+: Bidirectional mamba for time series forecasting. *arXiv preprint arXiv:2404.15772*.

Liu, S.; Yu, H.; Liao, C.; Li, J.; Lin, W.; Liu, A. X.; and Dustdar, S. 2022. Pyraformer: Low-Complexity Pyramidal Attention for Long-Range Time Series Modeling and Forecasting. In *International Conference on Learning Representations*.

Liu, Y.; Hu, T.; Zhang, H.; Wu, H.; Wang, S.; Ma, L.; and Long, M. 2024. iTransformer: Inverted Transformers Are Effective for Time Series Forecasting. In *The Twelfth International Conference on Learning Representations*.

Ma, S.; Kang, Y.; Bai, P.; and Zhao, Y.-B. 2024a. Fmamba: Mamba based on fast-attention for multivariate time-series forecasting. *arXiv preprint arXiv:2407.14814*.

Ma, S.; Zhang, T.; Zhao, Y.-B.; Kang, Y.; and Bai, P. 2023. TCLN: A Transformer-based Conv-LSTM network for multivariate time series forecasting. *Applied Intelligence*, 53(23): 28401–28417.

Ma, S.; Zhao, Y.-B.; Kang, Y.; and Bai, P. 2024b. Multivariate Time-Series Modeling and Forecasting With Parallelized Convolution and Decomposed Sparse-Transformer. *IEEE Transactions on Artificial Intelligence*, 5(10): 5232–5243.

Nie, Y.; H. Nguyen, N.; Sinthong, P.; and Kalagnanam, J. 2023. A Time Series is Worth 64 Words: Long-term Forecasting with Transformers. In *International Conference on Learning Representations*.

Qureshi, M.; Arbab, M. A.; and Rehman, S. u. 2024. Deep learning-based forecasting of electricity consumption. *Scientific Reports*, 14(1): 6489.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L. u.; and Polosukhin, I. 2017. Attention is All you Need. In Guyon, I.; Luxburg, U. V.; Bengio, S.; Wallach, H.; Fergus, R.; Vishwanathan, S.; and Garnett, R., eds., *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Wang, S.; Jiang, Y.; Li, Q.; and Zhang, W. 2024. Timely ICU Outcome Prediction Utilizing Stochastic Signal Analysis and Machine Learning Techniques with Readily Available Vital Sign Data. *IEEE Journal of Biomedical and Health Informatics*, 28(9): 5587–5599.

Wang, Z.; Kong, F.; Feng, S.; Wang, M.; Yang, X.; Zhao, H.; Wang, D.; and Zhang, Y. 2025. Is mamba effective for time series forecasting? *Neurocomputing*, 619: 129178.

Woo, G.; Liu, C.; Sahoo, D.; Kumar, A.; and Hoi, S. 2022. CoST: Contrastive Learning of Disentangled Seasonal-Trend Representations for Time Series Forecasting. In *International Conference on Learning Representations*.

Wu, H.; Xu, J.; Wang, J.; and Long, M. 2021. Autoformer: Decomposition Transformers with Auto-Correlation for Long-Term Series Forecasting. In Beygelzimer, A.; Dauphin, Y.; Liang, P.; and Vaughan, J. W., eds., *Advances in Neural Information Processing Systems*.

Yue, Z.; Wang, Y.; Duan, J.; Yang, T.; Huang, C.; Tong, Y.; and Xu, B. 2022. Ts2vec: Towards universal representation of time series. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, 8980–8987.

Zeng, A.; Chen, M.; Zhang, L.; and Xu, Q. 2023. Are transformers effective for time series forecasting? In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, 11121–11128.

Zhang, G. P. 2003. Time series forecasting using a hybrid ARIMA and neural network model. *Neurocomputing*, 50: 159–175.

Zhou, H.; Zhang, S.; Peng, J.; Zhang, S.; Li, J.; Xiong, H.; and Zhang, W. 2021. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, 11106–11115.

Zhou, T.; Ma, Z.; Wen, Q.; Wang, X.; Sun, L.; and Jin, R. 2022. FEDformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *International conference on machine learning*, 27268–27286. PMLR.