DeCoP: Dependency Controlled Pre-training for Time Series Representation Learning

YUEMIN WU

B.Eng

Supervisor: Dr. Chang Xu
Associate Supervisor: Dr. Daochang Liu

A thesis submitted in fulfilment of
the requirements for the degree of
Master of Philosophy

School of Computer Science
Faculty of Engineering
The University of Sydney
Australia

21 July 2025

DeCoP: Dependency Controlled Pre-training for Time Series Representation Learning

Yuemin Wu

Supervisor: Dr. Chang Xu

School of Computer Science

Faculty of Engineering

The University of Sydney

Copyright in Relation to This Thesis

Statement of Original Authorship

This is to certify that to the best of my knowledge, the content of this thesis is my own work. This thesis has not been submitted for any degree or other purposes. I certify that the intellectual content of this thesis is the product of my own work and that all the assistance received in preparing this thesis and sources have been acknowledged.

Signature:

Authorship attribution statement

I, Yuemin Wu, designed the study, analysed the data and wrote the drafts. In addition, the core content of this thesis is derived from this paper that has been submitted in the IEEE / CVF Computer Vision and Pattern Recognition Conference (CVPR) 2025. I designed and implemented the algorithm, completed the experiments, and accomplished proving the algorithm.

Student Name: Yuemin Wu
Date: 27 December 2024

As supervisor for the candidature upon which this thesis is based, I can confirm that the authorship attribution statements above are correct.

Supervisor Name: Chang Xu
Date: 27 December 2024

Abstract

Masked time series modeling (MTM) has become a leading approach in self-supervised pre-training for time series data. However, existing frameworks struggle to effectively model dependencies that balance informative signals and noise, often leading to overfitting or missing critical temporal dependencies due to non-stationarity and limited semantic context in time series data. To address this, we introduce **DeCoP**, a **D**ependency **Co**ntrolled **P**re-training framework that enhances self-supervised time series representation by effectively controlling dependency modeling, while significantly reducing computational cost. DeCoP controllably reduces non-stationary noise and disentangles mixed temporal variations from coarse to fine levels, thereby improving representation clarity. Specifically, DeCoP incorporates Instance-wise Patch Normalization (IPN) for controlled dependency, which reduces noise and stabilizes data distributions, establishing a controlled foundation for dependency modeling. Furthermore, a Hierarchical Dependency Controlled Learning (DCL) strategy is employed to selectively control inter-patch dependency ranges, generating robust, generalizable embeddings that enhance model stability across varying time series patterns. Extensive evaluations on ETTh1 datasets reveal that DeCoP achieves up to a 3% improvement in MSE over PatchTST, using only 37% of the FLOPs required.

Acknowledgements

This thesis could not have been completed without the help of my supervisors, fellow faculty members, and my friends and family.

First of all, I would like to express my heartfelt gratitude to my supervisors, Professor Chang Xu and Dr. Daochang Liu. Their invaluable guidance and support have been instrumental in navigating the uncertainties of my research journey. At the beginning of my studies, I was unsure of the direction I should take in my master's career. Professor Chang Xu provided me with invaluable advice and guidance, helping me to develop strong research values. With his support, I gradually discovered the direction I wanted to pursue and identified a meaningful project to dedicate myself to. Dr. Daochang Liu, on the other hand, consistently offered insightful ideas to tackle problems whenever I faced challenges. His encouragement and timely assistance helped me regain confidence in my research whenever I felt disheartened. Their support not only guided me through difficult moments but also strengthened my commitment to scientific exploration.

Secondly, I would like to express my deepest gratitude to senior colleagues, Dr Xiu Su. At the start of my graduate career, when I was unsure how to approach my research, he provided me with invaluable advice and guidance, showing me the path to becoming a competent research student. Under his guidance, I gradually came to understand the qualities of a good research idea and learned how to independently solve research problems. Additionally, his shared experiences in thesis writing significantly enhanced my writing skills, which have been instrumental in my academic progress. His support and wisdom have left a lasting impact on my journey as a researcher.

Thirdly, the completion of this thesis was greatly supported by the many energetic and outstanding researchers in my research group. Their company and support made my postgraduate journey both pleasant and fulfilling. They consistently brought fresh research ideas and broadened my perspective, significantly enriching my academic experience. Specifically,

I would like to thank Anh-Dung Dinh, Chen Chen, Chengbin Du, Chuyang Zhou, Jinxu Lin, Linwei Tao, Tao Huang, Xiaohuan Pei, Xiyu Wang, Yanxiang Ma, Zijian Wang, and Zunzhi You (listed alphabetically in no particular order), and many other colleagues whose names are not listed.

Fourthly, I would like to extend my heartfelt gratitude to my family for their endless support in helping me pursue my dreams. I am deeply thankful to my mother, Tiying Wu, my father, Jianjiang Wu, my mother-in-law, Ping Yu, my father-in-law, Ze Zhu, and my beloved partner, Younan Zhu. Your care, encouragement, insightful advice on both academics and life, and unwavering support throughout these years have been invaluable to me. I am truly fortunate to have you by my side.

Contents

CHAPTER 1

Introduction

Time series analysis is crucial in applications like weather forecasting [1; 2; 3], fault detection [4; 5], and sales prediction [6; 7]. With abundant unlabeled time series data across domains, pre-training approaches for representation learning without extensive annotation are increasingly popular. Recent research has thus focused on foundational models to address tasks like forecasting and classification in a general-purpose backbone [8; 9; 10; 11].

Despite these efforts, a foundational model for time series data comparable to GPT [12] or LLama [13] is still lacking, primarily due to two challenges: (1) inherent distribution shifts complicate pre-training in limited data conditions [14; 15; 16]. Publicly available time series data (about 1 million instances) is minimal compared to the trillions in text or image datasets, making adaptation to distributional shifts difficult [17; 18]; and (2) individual time series lack the rich semantic structure found in text or images [19; 20], where meaningful context can be derived from phrases or patches. In contrast, time series data is sparse and lacks semantic richness at individual points.

Based on these challenges, recent studies have explored specialized pre-training frameworks for time series, particularly masked time series modeling (MTM) [21; 22; 18] and contrastive learning (CL) [5]. While CL excels at capturing global semantic information, it has recently been shown to underperform MTM on both low-level prediction and high-level classification tasks [21; 22; 18; 5]. Furthermore, although MTM-based approaches address non-stationarity by incorporating Instance Normalization (IN) [15], IN often over-smooths the data—obscuring key features such as peaks and valleys. These observations underscore the need for methods that effectively stabilize data distributions without sacrificing critical temporal information.

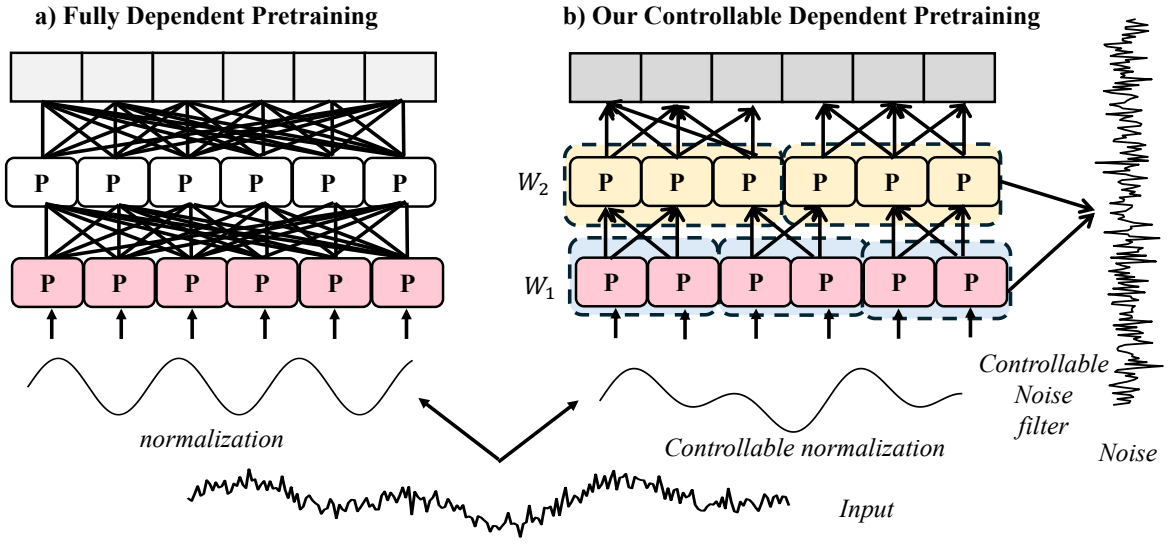

FIGURE 1.1. Comparison of uncontrolled Fully Dependent Pre-training (FDP) framework (a) and my Dependency Controlled Pre-training (DeCoP) framework (b). DeCoP uses controlled normalization and noise filtering to selectively capture dependency ranges, reducing redundancy and enhancing representation clarity.

Additionally, current frameworks fail to model the complex semantic dependencies in time series, limiting their ability to capture essential temporal relationships. They typically rely on either fully dependent patch modeling (e.g., transformers) [22] as shown in Figure 1.1a or fully independent patch modeling (e.g., linear structures) [21]. However, fully independent models risk losing valuable dependencies within subsequences [23; 24; 25], while fully dependent models often capture redundant information, especially in sparse time series [26; 27]. Consequently, existing frameworks lack an effective mechanism to balance informative signals and noise, essential for controlled dependency modeling. This limitation often leads to overfitting or loss of crucial temporal dependencies, underscoring the need for a balanced, controlled dependency modeling framework for diverse time series patterns.

In this paper, we introduce **DeCoP**, a **D**ependency **Co**ntrolled **P**re-training framework designed to enhance time series representation through controlled dependency modeling. DeCoP effectively balances informative signals and noise, addressing limitations in traditional fully dependent models by reducing overfitting and capturing critical temporal dependencies, as shown in Figure 1.1b. Specifically, DeCoP incorporates Instance-wise Patch Normalization

(IPN), which mitigates non-stationary noise and stabilizes data distributions, providing a controlled foundation for dependency modeling.

Building on this foundation, we introduced a Hierarchical Dependency Controlled Learning (DCL) strategy that selectively captures inter-patch dependencies by adjusting dependency ranges across multiple scales, progressively disentangling temporal variations from coarse to fine levels. Furthermore, we designed Frequency-domain Filtering (FDF) module that filters low-amplitude frequencies to generate denoised positive samples, enhancing model generalization across diverse time series patterns. This controlled pre-training process leverages both reconstruction and contrastive losses to guide effective representation learning, as illustrated in Figure 3.1b. The proposed method achieves state-of-the-art (SOTA) performance on the ETTh1 dataset, with up to a 3% improvement in forecasting MSE over PatchTST [22].

The main contributions of this work are as follows:

- We propose DeCoP, a dependency-controlled pre-training framework that enhances time series representation by balancing informative signals and noise, addressing challenges of non-stationarity and limited context.
- We develop an Instance-wise Patch Normalization for dependency learning, which reduces non-stationary noise and stabilizes data distributions, establishing a controlled foundation for dependency modeling in time series data.
- DeCoP incorporates a Dependency Controlling Pretraining strategy that selectively captures inter-patch dependencies across multiple scales and generates robust, generalizable embeddings, enabling DeCoP to effectively manage temporal dependencies.
- Extensive experiments on ETTh1 datasets show that DeCoP achieves up to 3% improvement in forecasting MSE over PatchTST, while requiring only 37% FLOPs.

CHAPTER 2

Literature review

In this chapter, we will first review the literature on self-supervised representation learning, followed by a review of the literature on two related downstream tasks: time series forecasting and classification.

2.1 Self-Supervised Learning

Self-supervised learning (SSL) has emerged as a dominant paradigm across domains, with notable methodologies such as Masked Language Modeling (MLM) [28], Generative Pre-trained Models (GPM) [29], and contrastive learning [30; 31]. These approaches leverage large-scale unlabeled datasets to learn meaningful representations without requiring manual labeling, enabling scalable learning while preserving data diversity and minimizing labeling costs. This section provides an overview of key SSL methods, including their adaptation to time series data.

2.1.1 Masked Time Series Learning

Masked Time Series Modeling (MTM) involves masking random segments of a time series and predicting them based on surrounding unmasked segments. This approach was inspired by the seminal BERT model [28], which pioneered masked token prediction in the Natural Language Processing (NLP) domain. In the time series domain, PatchTST [22] introduced a similar training paradigm by dividing time series into small patches and adopting a channel independence strategy that treats features independently. This design draws an analogy between time series patches and sentences in NLP, enabling the use of transformer blocks

to model dependencies across patches. Building upon this, SimMTM [18] highlighted the limitations of masking, which can disrupt key semantic information in time series data, such as trends, periodicity, and peaks. To address this, they leveraged manifold theory to generate positive samples and reconstructed time series from multiple masked sequences, preserving essential temporal characteristics. Recently, PITS [21] proposed a novel approach that avoids reconstructing masked segments. Instead, they reconstructed unmasked segments independently using only Multi-Layer Perceptrons (MLPs) during pre-training. This approach improved generalization performance in downstream tasks compared to traditional masked learning methods. In contrast, our framework better at handling complex distributions and noise by incorporating dependence controlled learning.

2.1.2 Contrastive Learning

Contrastive Learning (CL) [32; 33] has gained significant attention due to its effectiveness in maximizing similarity between positive pairs while minimizing similarity with negative pairs. Foundational works such as SimCLR [30] and MoCo [31] in computer vision, and CLIP [34] in multimodal alignment, demonstrate its versatility. In the time series domain, TF-C [5] introduced a time and frequency domain contrastive learning framework to enhance consistency across these domains. Similarly, TS2Vec [20] conducted contrastive learning along instance-wise and temporal dimensions to capture multi-scale contextual information. CoST [35] focused on time and frequency domain representations, leveraging a carefully designed contrastive loss to ensure causal consistency.

2.1.3 Large Time Series Foundation Models

Inspired by the success of GPT [29], researchers have begun exploring the development of foundation models (FMs) for time series data. Timer [10] employs a decoder-only architecture to predict future tokens in an autoregressive manner. Moment [8] and Moirai [11] adopt transformer encoder structures for pre-training large time series models on specific datasets, followed by fine-tuning for downstream tasks. However, the field lacks a general-purpose

backbone model akin to GPT for time series. This limitation stems from the absence of large-scale, publicly available time series datasets comparable to those in Computer Vision (CV) or NLP. Consequently, the sizes of time series foundation models remain relatively small, limiting their generalization capabilities. Furthermore, some pre-trained large time series models are tailored for specific tasks, such as forecasting [36; 11], rather than serving as versatile models across a range of applications.

2.2 Time Series Forecasting

Time series forecasting (TSF) involves predicting future values of a time series based on historical observations. Formally, given a sequence of past observations $\mathbf{X}_{1:T} = (x_1, x_2, \ldots, x_T) \in \mathbb{R}^T$ over T time steps, the goal is to forecast the next H time steps: $\hat{\mathbf{X}}_{T+1:T+H} = f_\theta\left(\mathbf{X}_{1:T}\right) \in \mathbb{R}^H$. Here, f_θ denotes the forecasting model, where T is the context length, and H represents the forecasting horizon. Traditionally, TSF has relied on task-specific learning, where models are tailored for individual datasets, such as stock prices or weather. Informer [37] introduced sparse attention to reduce the computational complexity for large input lengths T. Autoformer [38] proposed a decomposition architecture to separate trend and seasonal components, mitigating distribution shifts in time series. FEDformer [39] extended this by leveraging Fourier and Wavelet transforms, enabling better global feature extraction with lower computational costs. Despite their advantages, transformer-based models often face overfitting issues. DLinear [40] addressed this with a simple linear model, achieving superior performance with fewer parameters. Building on this, iTransformer [41] inverted the roles of transformer components to enhance multivariate correlations through variate token attention. This design improves scalability with larger lookback windows, addressing the performance degradation common in traditional transformer-based TSF models. More recently, inspired by the seasonal and trend decomposition strategies, CycleNet [42] introduced a novel approach that employs learnable parameters to directly model the seasonality of each dataset. This method enables CycleNet to effectively capture periodic patterns and achieve SOTA performance on datasets with clear seasonal components. However, a notable limitation of CycleNet is its inability to handle time series data that lack obvious periodicity, which restricts its

applicability to a broader range of forecasting tasks. Instead of learning a task-specific model, our framework try to learning a general purpose pre-training models which can improve the performance of forecasting with less parameters and training time.

2.3 Time Series Classification

Time series classification (TSC) involves assigning a categorical label to a time series instance based on its temporal patterns. Formally, a dataset $D = (X_1, Y_1), (X_2, Y_2), ..., (X_N, Y_N)$ is a collection of N pairs, where X_i represents a univariate or multivariate time series, and Y_i is its corresponding label. For datasets with K classes, Y_i is is typically represented as a one-hot vector of length K, where each element $j \in [1, K]$ is equal to 1 if the class of X_i is j, and 0 otherwise. TSC is a high-level learning task which need capture the global semantic relationship between time series, and currently contrastive learning is the mainstream paradigm in TSC task. Except for the methods mentioned in section 2.1.2, TS-TCC [43] perform contrastive learning by controlling the intensity of augmentation function, and them learn different view of data in both time and contextual aspect. CLUDA [44] leverage nearest-neighborhood CL to learn contextual representation by only create positive samples. Unlike previous CL model, our framework combines the self-supervised nature of MTM with the contrastive principles of CL, enhancing robustness and consistency in feature learning to address distribution shifts.

2.4 Chapter Conclusion

This chapter reviewed the SOTA methods and challenges in self-supervised learning (SSL) and its applications to time series forecasting (TSF) and classification (TSC). SSL has emerged as a dominant paradigm across domains, with approaches such as Masked Language Modeling (MLM), Generative Pre-trained Models (GPM), and contrastive learning enabling scalable representation learning from large, unlabeled datasets. Specific adaptations to time series data, such as Masked Time Series Modeling (MTM), have shown promise by leveraging ideas from natural language processing, with models like PatchTST addressing limitations of

traditional masking-based methods. Meanwhile, contrastive learning techniques, including TS2Vec and CoST, have demonstrated effectiveness in capturing temporal and frequency domain consistency.

For TSF, the literature highlighted a shift from task-specific models to general-purpose frameworks. Models like Informer and Autoformer have tackled challenges of long input sequences and distribution shifts using sparse attention and decomposition architectures. FEDformer and Cyclenet further refined these methods by integrating Fourier and Wavelet transforms and learnable parameters for seasonality modeling. However, limitations such as handling non-periodic data and scalability issues remain prominent.

In TSC, contrastive learning has become the mainstream approach, with methods like TS-TCC and CLUDA leveraging augmented views and contextual representation learning. These advances emphasize the importance of capturing global semantic relationships in time series data. The review concludes that while existing methods offer significant advancements, challenges such as handling complex distributions, noise, and diverse temporal patterns persist, motivating the development of more robust and generalizable frameworks.

CHAPTER 3

Methods

3.1 MTM Preliminaries

Most existing time series modeling follow the same transformer structure in Natural Language Processing (NLP) or Computer Vision (CV), as shown in 3.1a. Given an univariate time series $\mathcal{X} \in \mathbb{R}^L$ with look-back length L, the output is masked time series patch $\hat{Y}$ and the model is optimized by reconstructing the randomly masked patches by MSE $= \|\mathbf{y} - \hat{\mathbf{y}}\|_2^2$. The overall pipeline is as follows:

$$\mathcal{X} = Mask(Patch(Norm(\mathcal{X}))). \tag{3.1}$$

In this process, instance normalization is applied to calculate the mean μ and variance σ over the past L time points, ensuring the data is standardized. Subsequently, a patching operation is employed to divide the time series into "tokens," analogous to tokenization in large language models. These tokens are then passed into self-attention blocks to capture and model semantic relationships:

$$\mathbf{Z}_h = Softmax\left(\frac{\mathbf{Q}_h \cdot \mathbf{K}_h}{\sqrt{d_k}}\right)\mathbf{V}_h. \tag{3.2}$$

Where $\mathbf{Q}_h, \mathbf{K}_h, \mathbf{V}_h$ are computed through corresponding parameter matrices. Finally, a flattening operator is applied to obtain the representation of the past L time points, and a linear layer is used to reconstruct the masked patches:

$$\hat{Y} = LinearHead(Flatten(\mathbf{Z}_h)), \tag{3.3}$$

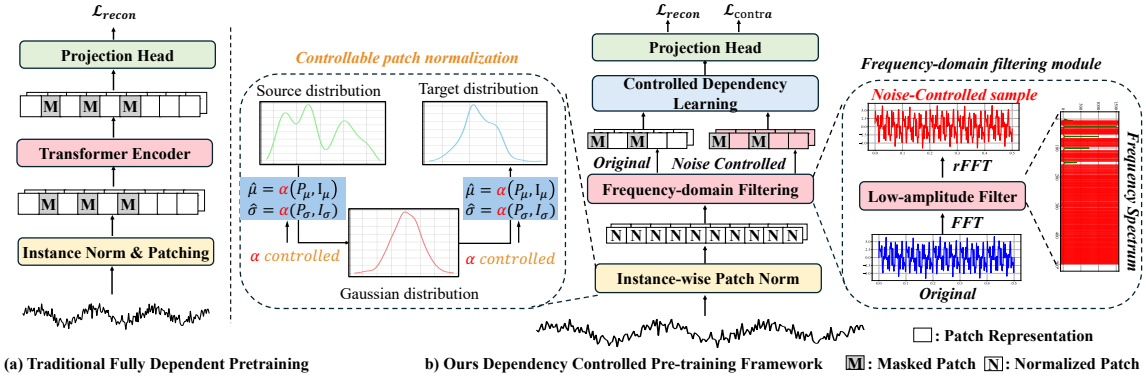

FIGURE 3.1. (a) Traditional fully dependent pre-training framework for time series. (b) The proposed **DeCoP** framework, featuring controlled dependency modeling through Instance-wise Patch Normalization (IPN), which reduces noise and stabilizes distributions, and a Frequency-domain filtering module (FDF) that filters low-amplitude frequencies for noise-robust samples. Hierarchical Dependency Controlled Learning (DCL) adjusts inter-patch dependency ranges across scales.

While this framework effectively restores masked patches, its fully dependent learning paradigm often amplifies additional noise, which can lead to overfitting and increased computational costs. To overcome these limitations, incorporating dependency-controlled learning has been proposed. This approach selectively captures dependencies, thereby reducing noise and computational demands while simultaneously producing more generalized and robust representations.

3.2 Method

In this section, we introduce the **DeCoP** framework, which enhances self-supervised time series representation via controlled dependency modeling (see Fig.3.1b). DeCoP comprises two main parts: (a) Instance-wise Patch Normalization (see section 3.2.1), which reduces noise and stabilizes distributions; and (b) Dependency Controlled Pre-training (see section 3.2.2 and section 3.2.3), which selectively captures inter-patch dependencies ranges across scales. This approach improves robustness by filtering noise and enabling precise dependency adjustments, allowing DeCoP to manage temporal dependencies effectively.

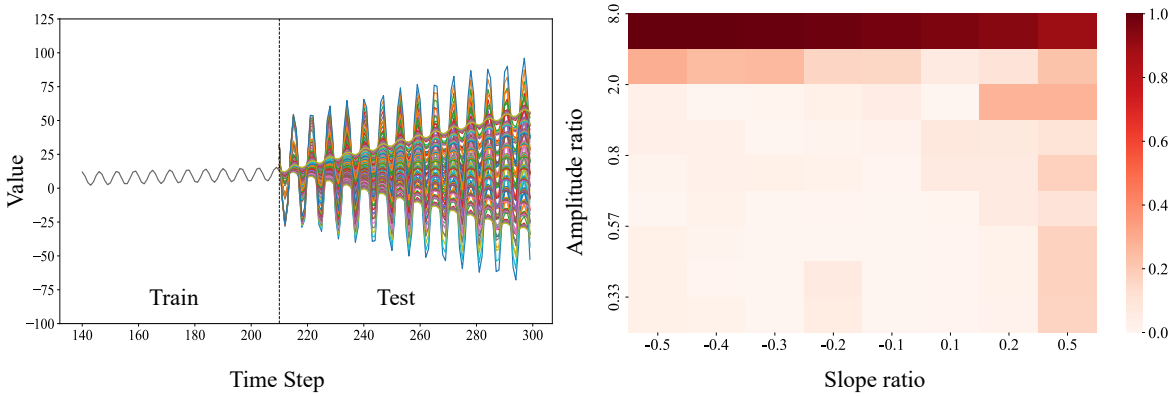

FIGURE 3.2. A toy example illustrating distribution shift and MSE differences between models with and without Instance-wise patch normalization. Darker colors indicate larger differences, highlighting the effectiveness of instance-wise patch normalization.

3.2.1 Instance-wise Patch Normalization for Controlled Dependency

To address the challenge of distribution shifts and maintain the unique characteristics of each patch during dependency-controlled learning, we utilize instance-wise patch normalization. This method leverages intra-patch information to adaptively learn the mean μ and standard deviation θ of each time series, as illustrated in Figure 3.1b. Specifically, for a univariate time series $\mathcal{X}$, defined with patch length P and stride S. the sequence is divided into N patches as follows:

$$\mathcal{X} = \{\mathbf{x_1}, \mathbf{x_2}, ..., \mathbf{x_N}\}, N = \left\lfloor \frac{L-P}{S} \right\rfloor + 2. \tag{3.4}$$

Where N is the number of patches. To highlight intra-patch variation, we then calculate mean and standard deviation at the patch-wise scale:

$$\mathbb{E}_p\left[\mathbf{x_N}\right] = \frac{1}{P}\sum_{i=1}^{P} x_{p,i}, \tag{3.5}$$

Where i is the index of each time points within each patch. For each patch, we calculate the mean and variance along the P dimension.

$$Var_p\left[\mathbf{x_N}\right] = \frac{1}{P}\sum_{i=1}^{P}\left(x_{p,i} - \mathbb{E}_t\left[x_{p,t}\right]\right)^2. \tag{3.6}$$

To also incorporate global distribution information, we leverage normalization at instance granularity as follows:

$$\mathbb{E}_t\left[\mathcal{X}\right] = \frac{1}{L}\sum_{j=1}^{L} x_{tj}, \tag{3.7}$$

where j is the relative index of time series $\mathcal{X}$. For each time series, mean and variance are calculated along L dimension.

$$Var_t\left[\mathcal{X}\right] = \frac{1}{L}\sum_{j=1}^{L}\left(x_{tj} - \mathbb{E}_t\left[\mathcal{X}\right]\right)^2. \tag{3.8}$$

For more stable training of controlled dependency modeling, a hyper-parameter $\alpha \in \mathbb{R}$ is used to adjust the weight of each statistic, controlling the influence of each component as follows:

$$\mathbb{E}\left[\mathbf{x_N}\right] = (1-\alpha) \times \mathbb{E}_t + \alpha \times \mathbb{E}_p, \tag{3.9}$$

where $\mathbb{E}_t$ and $\mathbb{E}_p$ are the global mean and local mean of current patch and $\mathbf{x_N}$.

$$Var\left[\mathbf{x_N}\right] = (1-\alpha) \times Var_t + \alpha \times Var_p. \tag{3.10}$$

Finally, the input data $\mathbf{x_N}$ is transformed into $\hat{\mathbf{x}}_\mathbf{p}$ by applying calculated mean and variance through normalization:

$$\hat{\mathbf{x}}_\mathbf{p} = \frac{\mathbf{x_N} - \mathbb{E}\left[\mathbf{x_N}\right]}{\sqrt{\mathrm{Var}\left[\mathbf{x_N}\right] + \epsilon}}. \tag{3.11}$$

where ϵ is a small constant to prevent division by zero. By this way, we create a stable foundation for dependency modeling, which can overcome distribution shift problem effectively as shown in Fig.3.2.

3.2.2 Frequency-domain Filtering for Robust Representation Learning

To learn more robust representation model, we introduce a data augmentation method based on Fourier transformation to filter the noisy spectrum as illustrated in Fig.3.3a and Fig.3.4 and a complementary contrastive learning loss. By combining Frequency-domain Filtering (FDF) with a complementary contrastive learning loss, the method enhances the generalization

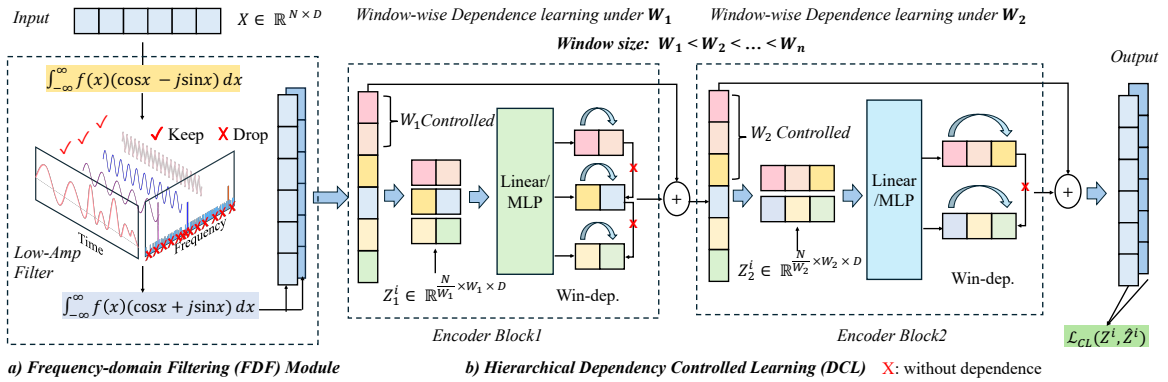

FIGURE 3.3. (a) Frequency-domain Filtering (FDF) module, which generates noise-robust positive samples by retaining relevant frequencies and discarding low-amplitude ones. (b) The Hierarchical Dependency Controlled Learning (DCL) module with two encoder blocks, applying window-wise dependency learning (Win-dep) at different scales. This approach enables adaptive inter-patch correlation modeling across varying window sizes, capturing dependencies at multiple temporal resolutions.

capability of the model. Technically, we utilize the amplitude information in frequency domain and filter top-k low-amplitude frequency to generate positive samples $(\mathcal{X}_p, \tilde{\mathcal{X}}_p)$. The overall process is as follows:

Given an input sequence $\mathcal{X}_p$, we perform Fourier transform to obtain the corresponding frequency spectrum,

$$\mathcal{F}(\mathcal{X}_p) = \int_{-\infty}^{\infty} \mathcal{X}_p \, e^{-j2\pi ft} \, dt. \tag{3.12}$$

where the resulting spectrum $S = 0, 1, ..., [L/2]$. A controlled low-amplitude filter is then applied to mask low-energy components, ensuring the retention of key frequencies while suppressing noise:

$$\mathcal{S}(\mathcal{X}_p) = Mask\left(Filter\left(topK, \mathcal{F}(\mathcal{X}_p)\right)\right), \tag{3.13}$$

where $TopK = \beta \times L$, and β is a hyper-parameter to control the intensity of filter. After that, we conduct Inverse Fourier Transform $\mathcal{F}^{-1}$ to obtain de-noised sample $\tilde{\mathcal{X}}_p$.

$$\tilde{\mathcal{X}}_p = \mathcal{F}^{-1}\left(\mathcal{X}_p\right) = \int_{-\infty}^{\infty} \mathcal{S}(\mathcal{X}_p) \, e^{-j2\pi ft} \, d\mathcal{X}_p, \tag{3.14}$$

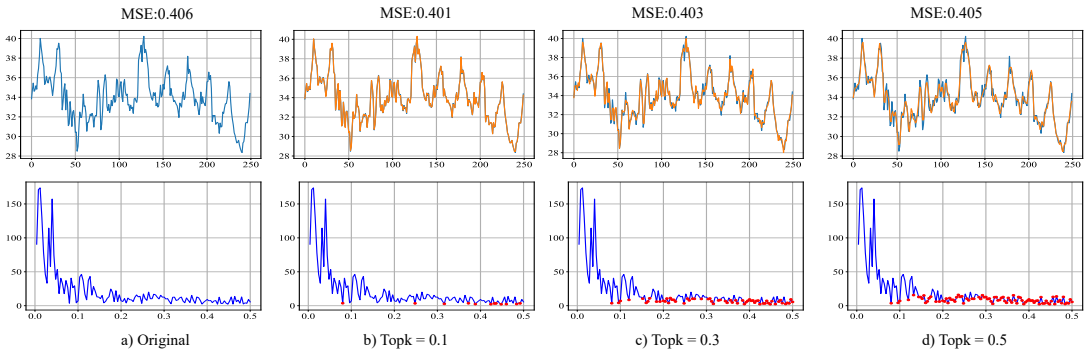

FIGURE 3.4. Effect of low-amplitude filtering on the ETTh1 dataset with varying Top-k values. The first row shows the waveform, and the second row displays the amplitude spectrum, with red dots indicating filtered frequencies. The Top-k controls filtering intensity: (a) Original signal (no filtering); (b) Top-k = 0.1, filtering the lowest 10% of frequencies; (c) Top-k = 0.3, filtering 30%; (d) Top-k = 0.5, filtering 50%.

For each positive pairs $(\mathcal{X}_p, \tilde{\mathcal{X}}_p)$, we define a softmax-like contrastive loss function L_{cl} using the output of each encoder layer:

$$\mathcal{Z}_k = \mathcal{E}_{enc}(\mathcal{X}_p), \tilde{\mathcal{Z}}_k = \mathcal{E}_{enc}(\tilde{\mathcal{X}}_p), \tag{3.15}$$

where $\mathcal{E}_{enc}$ is the controlled encoder we will introduce in section 3.2.3. The similarity for each positive series pair is computed as:

$$\mathcal{P}(\mathcal{Z}_k, \tilde{\mathcal{Z}}_k) = \frac{\exp(\mathcal{Z}_k \cdot \tilde{\mathcal{Z}}_k)}{\sum_{\bar{\mathbf{z}}_\mathbf{k} \neq \tilde{\mathbf{z}}_\mathbf{j}} \exp(\mathcal{Z}_k \cdot \tilde{\mathcal{Z}}_k)}. \tag{3.16}$$

By minimize L_{cl}, the noised $\mathcal{X}_p$ will have more accurate representation like de-noised $\tilde{\mathcal{Z}}_k$.

$$L_{cl} = -\frac{1}{BMK} \sum_{i=1}^{B} \sum_{m=1}^{M} \sum_{k=1}^{K} \log \mathcal{P}(\mathcal{Z}_k, \tilde{\mathcal{Z}}_k), \tag{3.17}$$

where, B is the batch size, M is the variable size and K is the number of encoder block.

3.2.3 Hierarchical Dependency Controlled Learning

To fully capture meaningful semantic information between patches while minimizing redundant noise, we propose a hierarchical dependency learning encoder with controlled windows,

where inter-patch correlation is progressively learned through multi-scale windows at different layers, as shown in Figure 3.3b.

Technically, for de-noised positive pair$(\mathcal{X}_p, \tilde{\mathcal{X}}_p)$, where $\mathcal{X}_p \in \mathbb{R}^{P\times N}$. This module first project patches into latent space via linear layer:

$$\mathcal{Z}_p = \mathcal{W}_P\mathcal{X}_p + \mathcal{B}, \tag{3.18}$$

where $W_P \in \mathbb{R}^{P\times D}$, with D as the model dimension and $\mathcal{B}$ as the bias. Additionally, relative time position information is essential for controlled patch dependence learning due to inherent periodical characteristic. A learnable additive position encoding $W_{pos} \in \mathbb{R}^{\mathbb{D}\times\mathbb{N}}$ is applied to keep the temporal order:

$$\mathcal{Z}_p = \mathcal{Z}_p + W_{pos}. \tag{3.19}$$

Denoting different window size as $\{\mathcal{W}_k\}_{k=1}^{K}$, and each window contain different hidden size $\{\mathbf{d}_k\}_{k=1}^{K}$, where K is the number of encoder layer. Finally, we define a controlled learning block as follows:

$$\mathcal{Z}_r = Flatten(Reshape(Padding(\mathcal{Z}_p))). \tag{3.20}$$

For each layer, we first reshape $\mathcal{Z}_p$ into window size, from $\mathbb{R}^{N\times D}$ into $\mathbb{R}^{N/W_j,W_j,D}$, and then fatten the last two dimension as $\mathbb{R}^{N/W_j,W_j\times D}$. The high-level representation within controlled window can be modeled as follows:

$$\mathcal{Z}_j = \mathcal{E}_\theta(\mathcal{Z}_r), \tag{3.21}$$

where $\mathcal{E}_\theta$ is based on a linear or MLP structure in my experiment and finally we reshape the input into original size and leverage residual connection to ensure the training stability.

$$\mathcal{Z}_p = ReshapeBack(\mathcal{Z}_j) + Dropout(\mathcal{Z}_p) \tag{3.22}$$

The window size is increasing across each encoder layers and thus a global receptive view can be learned through carefully design.

3.2.4 Loss design

As illustrated in Figure 3.1b, the reconstruction of masked patches relies on the information from unmasked patches. A pre-training head is used to predict the masked patches, formulated as:

$$\hat{\mathcal{Z}}_p = W\mathcal{Z}_p, \tag{3.23}$$

where $W \in \mathbb{R}^{D\times P}$ represents the weight matrix, and $\mathcal{Z}_p \in \mathbb{R}^{N\times D}$. For the reconstruction process, we adopt the MSE as the reconstruction loss, defined as:

$$L_{recon} = \sum_{i=1}^{B}\sum_{m=1}^{M}\sum_{n=1}^{N} \left\|\mathrm{m} \circ (\mathbf{x}_p^{(i,m,n)} - \hat{\mathbf{x}}_p^{(i,m,n)})\right\|_2^2, \tag{3.24}$$

where B, M, and N represent the batch size, the number of feature size and the total number of patches, respectively. The mask indicator $\mathrm{m} = 1$ if this patch is masked, and $\circ$ denotes the element-wise product to ensure the loss is only calculated for masked patches. The final loss function combines the reconstruction loss L_{recon} with a contrastive loss L_{cl} weighted by a hyperparameter γ , as follows:

$$L = L_{recon} + \gamma \times L_{cl}, \tag{3.25}$$

where γ controls the contribution of the contrastive loss L_{cl}. This hybrid loss design ensures effective reconstruction of masked patches while simultaneously optimizing the model for contrastive representation learning.

CHAPTER 4

Results

In Section 4.1, we introduce the benchmark datasets and outline the implementation details of DeCoP and the baseline models. Next, Sections 4.2.1 and 4.2.2 present the experimental results on time series forecasting and classification in both in-domain and cross-domain settings, where DeCoP achieves SOTA performance under various metrics. Finally, we offer an extensive ablation and analysis study in Section 4.2.3, validating the effectiveness of each module in DeCoP.

4.1 Experiment

Experiment Setting. We conduct experiments on forecasting and classification tasks, following the protocols in [22] and [18]. Fine-tuning performance is evaluated under in-domain and cross-domain settings. MSE and MAE are used as metrics for forecasting, while Accuracy, Precision, Recall, and F1-score assess classification performance. For forecasting, six real-world datasets are employed, including four ETT datasets [37] (ETTh1, ETTh2, ETTm1, ETTm2), Weather [45], and Electricity [46]. The ETTh1, ETTh2, ETTm1, and ETTm2 were collected from two distinct electric transformers over a two-year period, from July 2016 to July 2018. The Weather dataset [45] comprises 21 meteorological indicators recorded every 10 minutes in Germany during 2020. The Electricity dataset [46] contains hourly electricity consumption records of 321 customers, spanning the years 2012 to 2014.

For classification, we use four real-world datasets: SleepEEG [47], Epilepsy [48], FD-B [49], and EMG [50]. SleepEEG includes 153 full-night EEG recordings across five sleep stages (W, N1–N3, REM). Epilepsy contains single-channel EEG from 500 subjects with binary seizure

TABLE 4.1. Datasets for Forecasting and Classification Tasks

Tasks	Datasets	Channels	Length	Classes	Frequency
Forecasting	ETTh1	7	17420	-	1 Hour
	ETTh2	7	17420	-	1 Hour
	ETTm1	7	69680	-	15 Mins
	ETTm2	7	69680	-	15 Mins
	Weather	21	52696	-	10 Mins
	Electricity	321	26304	-	1 Hour
Classification	SleepEEG	1	200	5	100 Hz
	Epilepsy	1	178	2	174 Hz
	FD-B	1	5120	3	64K Hz
	EMG	1	1500	3	4K Hz

TABLE 4.2. Forecasting Configuration

Task	Dataset	dmodel	Window Size	d1	d2
Forecasting	ETTh1/ETTh2	128	(2,5)	256	512
	ETTm1/ETTm2	128	(2,5)	256	512
	Weather	128	(2,5)	256	512
	Electricity	256	(3,6)	512	512

TABLE 4.3. Classification Configuration

Source Data	Target Data	dmodel	Window Size	d1	d2	Aggregate
Epilepsy	Epilepsy	128	(3,6)	256	512	avg
Epilepsy	Epilepsy	128	(2,5)	256	512	avg
SleepEEG	FD-B	128	(1,2)	128	256	avg
SleepEEG	EMG	64	(1,2)	64	64	max

labels. FD-B involves motor bearing signals classified into three fault types: undamaged, inner-damaged, and outer-damaged. EMG records muscle responses from three patients with neuropathy or myopathy, each representing a class. The characteristics are listed in Table 4.1.

4.1.1 Details of baseline settings

For the time series forecasting task, we categorize the baseline models into two paradigms: self-supervised and supervised. PatchTST [22] and SimMTM [18] are representative self-supervised models. In contrast, DLinear [40], FEDformer [39], Autoformer [38], and Informer [37] are robust supervised models for forecasting tasks. Additionally, CycleNet and Time-Mixer represent the latest SOTA methods, also grounded in the supervised paradigm. For the classification task, we divide the baseline models into two paradigms: masked time series models (MTM) and contrastive learning (CL) models. SimMTM [18], Ti-MAE [51], and

TST [25] follow the MTM paradigm, while LaST [52], TF-C [53], CoST [35], and TS2Vec [20] are based on the CL paradigm.

For time series forecasting task, the default look-back window for various MTM models is set to 512, following [22]. The results for DLinear , FEDformer, Autoformer, and Informer are from PatchTST. Meanwhile, for time series classification task,the results for Ti-MAE, TST , LaST , TF-C , CoST , and TS2Vec are obtained from SimMTM . For the latest SOTA methods [54; 42], the look-back window length is consistently fixed at 512, adhering to [22].

4.1.2 Implementation Details

All experiments were repeated five times, implemented using PyTorch, and conducted on an NVIDIA RTX 4090 GPU with 24GB of memory. The baselines were implemented based on their official repositories, adhering to the configurations specified in their original papers. For forecasting tasks, all datasets were chronologically split into training, validation, and test sets, with splitting ratios of 6:2:2 for the ETT datasets and 7:1:2 for the other datasets, as outlined in [38]. For classification tasks, the dataset splits followed the setup described in [53]. During pre-training, each model was typically trained for 100 epochs. This was followed by linear probing of the head for 10 epochs and fine-tuning the entire model for 20 epochs.

4.1.3 Model Parameters

By default, all experiments are configured with the following parameters: $e_{layers} = 2$, top$\mathcal{K} = 0.1$, $\alpha = 0.01$, $lr = 1e-4$ and $\gamma = 0.01$. During pre-training, a dropout ratio of 0.2 is applied. For forecasting tasks, both in-domain and cross-domain experiments share the same configuration, with a patch size and stride of 12. Additional key parameters for forecasting are detailed in Table 4.2. For classification tasks, the patch size is set to 8 for the Epilepsy dataset, while it remains 12 for other datasets, as in forecasting tasks. To address potential overfitting in smaller datasets, such as EMG, a reduced parameter size (dim = 64) is employed, as outlined in Table 4.3.

TABLE 4.4. Results of long-term forecasting tasks for the in-domain setting of forecasting the future $F \in \{96, 192, 336, 720\}$ time points . The best and the second best results are denoted by yellow and pink.

Models		**DeCoP/Linear**		**DeCoP/MLP**		SIMMTM [18]		PatchTST [22]		CycleNet [42]		TimeMixer [54]		Dlinear [40]		iTransformer [41]		Fedformer [39]		Autoformer [38]		Informer [37]	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	96	0.361	0.390	0.364	0.392	0.367	0.402	0.366	0.397	0.383	0.408	0.378	0.406	0.375	0.399	0.400	0.425	0.376	0.415	0.435	0.446	0.941	0.769
	192	0.394	0.410	0.400	0.416	0.401	0.425	0.431	0.443	0.410	0.426	0.444	0.448	0.405	0.426	0.427	0.443	0.423	0.446	0.456	0.457	1.007	0.786
	336	0.414	0.427	0.414	0.430	0.415	0.430	0.45	0.456	0.440	0.444	0.424	0.444	0.439	0.443	0.457	0.465	0.444	0.462	0.486	0.487	1.038	0.784
	720	0.437	0.458	0.451	0.463	0.43	0.453	0.472	0.484	0.487	0.482	0.485	0.485	0.472	0.490	0.631	0.574	0.469	0.492	0.515	0.517	1.144	0.857
	Avg	0.401	0.421	0.407	0.425	0.404	0.428	0.43	0.445	0.430	0.440	0.432	0.446	0.423	0.440	0.479	0.477	0.428	0.454	0.473	0.477	1.033	0.799
ETTh2	96	0.268	0.333	0.271	0.334	0.288	0.347	0.284	0.343	0.299	0.355	0.280	0.352	0.289	0.353	0.299	0.359	0.332	0.374	0.332	0.368	1.549	0.952
	192	0.327	0.373	0.333	0.377	0.346	0.385	0.355	0.387	0.354	0.394	0.367	0.400	0.383	0.418	0.377	0.406	0.407	0.446	0.426	0.434	3.792	1.542
	336	0.353	0.397	0.356	0.400	0.363	0.401	0.379	0.411	0.392	0.425	0.385	0.420	0.448	0.465	0.429	0.442	0.4	0.447	0.477	0.479	4.215	1.642
	720	0.383	0.426	0.398	0.439	0.396	0.431	0.4	0.435	0.424	0.451	0.469	0.480	0.605	0.551	0.444	0.466	0.412	0.469	0.453	0.490	3.656	1.619
	Avg	0.333	0.382	0.34	0.387	0.348	0.391	0.355	0.394	0.367	0.406	0.375	0.413	0.431	0.447	0.387	0.418	0.388	0.434	0.422	0.443	3.303	1.439
ETTm1	96	0.283	0.340	0.280	0.338	0.299	0.354	0.288	0.345	0.305	0.361	0.306	0.358	0.299	0.343	0.311	0.366	0.326	0.390	0.510	0.492	0.626	0.560
	192	0.337	0.367	0.323	0.365	0.343	0.379	0.330	0.372	0.343	0.379	0.347	0.381	0.335	0.365	0.348	0.385	0.365	0.415	0.514	0.495	0.725	0.619
	336	0.369	0.387	0.353	0.385	0.375	0.401	0.359	0.392	0.384	0.411	0.437	0.369	0.386	0.389	0.380	0.405	0.392	0.425	0.51	0.492	1.005	0.741
	720	0.423	0.416	0.409	0.418	0.431	0.439	0.406	0.421	0.439	0.431	0.466	0.425	0.424	0.422	0.443	0.444	0.446	0.458	0.527	0.493	1.133	0.845
	Avg	0.353	0.377	0.341	0.376	0.362	0.393	0.346	0.383	0.368	0.395	0.389	0.383	0.361	0.380	0.371	0.400	0.382	0.422	0.515	0.493	0.872	0.691
ETTm2	96	0.163	0.252	0.163	0.254	0.176	0.27	0.164	0.256	0.180	0.266	0.173	0.263	0.167	0.260	0.179	0.273	0.18	0.271	0.205	0.293	0.355	0.462
	192	0.218	0.290	0.219	0.293	0.232	0.304	0.223	0.296	0.234	0.302	0.228	0.301	0.224	0.303	0.242	0.315	0.252	0.318	0.278	0.336	0.595	0.586
	336	0.271	0.324	0.268	0.326	0.288	0.339	0.277	0.332	0.285	0.340	0.277	0.332	0.281	0.342	0.291	0.345	0.324	0.364	0.343	0.379	1.27	0.871
	720	0.364	0.385	0.346	0.376	0.381	0.396	0.365	0.387	0.371	0.392	0.369	0.390	0.397	0.421	0.377	0.398	0.41	0.420	0.414	0.419	3.001	1.267
	Avg	0.254	0.313	0.249	0.312	0.269	0.327	0.257	0.318	0.267	0.325	0.262	0.322	0.267	0.332	0.272	0.333	0.292	0.343	0.310	0.357	1.305	0.797
Weather	96	0.169	0.222	0.145	0.193	0.152	0.206	0.144	0.193	0.148	0.202	0.149	0.202	0.176	0.237	0.168	0.220	0.238	0.314	0.249	0.329	0.354	0.405
	192	0.213	0.258	0.190	0.237	0.197	0.246	0.190	0.236	0.192	0.244	0.198	0.246	0.220	0.282	0.209	0.254	0.275	0.329	0.325	0.370	0.419	0.434
	336	0.259	0.293	0.242	0.278	0.246	0.285	0.244	0.280	0.243	0.282	0.245	0.286	0.265	0.319	0.266	0.295	0.339	0.377	0.351	0.391	0.583	0.543
	720	0.325	0.340	0.314	0.329	0.314	0.335	0.320	0.335	0.314	0.333	0.321	0.340	0.323	0.362	0.341	0.345	0.389	0.409	0.415	0.426	0.916	0.705
	Avg	0.242	0.278	0.223	0.259	0.227	0.268	0.225	0.261	0.224	0.265	0.228	0.269	0.246	0.300	0.246	0.278	0.310	0.357	0.335	0.379	0.568	0.522
Electricity	96	0.137	0.234	0.127	0.222	0.133	0.223	0.126	0.221	0.126	0.221	0.135	0.234	0.140	0.237	0.132	0.227	0.186	0.302	0.196	0.313	0.304	0.393
	192	0.152	0.248	0.146	0.236	0.147	0.237	0.145	0.238	0.144	0.238	0.152	0.247	0.153	0.249	0.153	0.248	0.197	0.311	0.211	0.324	0.327	0.417
	336	0.168	0.263	0.161	0.256	0.166	0.265	0.164	0.256	0.160	0.255	0.169	0.268	0.169	0.267	0.168	0.264	0.213	0.328	0.214	0.327	0.333	0.422
	720	0.208	0.297	0.195	0.289	0.203	0.297	0.193	0.291	0.201	0.294	0.203	0.297	0.203	0.301	0.193	0.286	0.233	0.344	0.236	0.342	0.351	0.427
	Avg	0.166	0.260	0.157	0.251	0.162	0.256	0.157	0.252	0.158	0.252	0.165	0.261	0.166	0.264	0.161	0.256	0.207	0.321	0.214	0.327	0.329	0.415

4.2 Results and Discussion

4.2.1 Time series Forecasting

In-domain Evaluation. We compare our model with six competitive SOTA baseline methods in time series forecasting, including self-supervised approaches SimMTM [18], PatchTST [22]) and supervised approaches (CycleNet [42], TimeMixer [54], DLinear [40], iTransformer [41], Fedformer [39], Autoformer [38], Informer [37]). As shown in Table 4.4, DeCoP/Linear outperforms the second-best by 15% on ETTh2. For more complex datasets like weather, DeCoP/MLP achieves the best results, surpassing the second-best by 0.9% in MSE and MAE.

Cross-domain Analysis. In the cross-domain setting, we compare our framework with six advanced time series pre-training frameworks (SimMTM, PatchTST, TF-C [53], LaST [52], CoST [35] and TimeMAE [51]). As shown in Table 4.5, we evaluate multiple scenarios to test

TABLE 4.5. Complete results of long-term forecasting tasks are presented for the cross-domain setting. The best and the second best results are denoted by yellow and pink.

Cross-domain	Length	DeCoP-Linear		DeCoP-MLP		PatchTST[22]		SimMTM[18]		TF-C[53]		LaST [52]		Ti-MAE [51]		CoST [35]		TST [25]		TS2Vec [20]	
		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTm2 → ETTh1	96	0.362	0.391	0.367	0.393	0.385	0.411	0.388	0.421	0.968	0.738	0.428	0.454	0.433	0.431	0.403	0.426	0.389	0.413	0.483	0.48
	192	0.397	0.412	0.402	0.416	0.425	0.439	0.419	0.423	1.08	0.801	0.427	0.497	0.474	0.458	0.457	0.468	0.463	0.452	0.579	0.537
	336	0.421	0.429	0.429	0.436	0.44	0.451	0.435	0.444	1.091	0.824	0.528	0.54	0.515	0.448	0.794	0.682	0.492	0.465	0.673	0.563
	720	0.437	0.459	0.472	0.478	0.482	0.488	0.468	0.474	1.226	0.893	0.527	0.537	0.496	0.488	0.739	0.617	0.468	0.468	0.729	0.62
	Average	0.404	0.423	0.417	0.431	0.433	0.447	0.428	0.441	1.091	0.814	0.503	0.507	0.464	0.456	0.598	0.548	0.453	0.45	0.616	0.55
ETTh2 → ETTm1	96	0.305	0.348	0.283	0.340	0.302	0.353	0.322	0.347	0.677	0.603	0.314	0.396	0.323	0.362	0.322	0.351	0.338	0.383	0.679	0.546
	192	0.339	0.368	0.328	0.369	0.342	0.375	0.332	0.375	0.718	0.638	0.587	0.545	0.37	0.395	0.331	0.373	0.394	0.408	0.673	0.551
	336	0.368	0.385	0.354	0.388	0.37	0.392	0.394	0.391	0.755	0.663	0.631	0.584	0.397	0.413	0.382	0.397	0.401	0.412	0.703	0.557
	720	0.423	0.415	0.413	0.420	0.439	0.426	0.411	0.424	0.848	0.712	0.368	0.429	0.442	0.439	0.417	0.428	0.434	0.432	0.722	0.573
	Average	0.359	0.379	0.345	0.379	0.363	0.387	0.365	0.384	0.75	0.654	0.475	0.489	0.383	0.402	0.363	0.387	0.391	0.409	0.694	0.557
ETTm1 → ETTh1	96	0.367	0.395	0.370	0.394	0.388	0.411	0.367	0.398	0.666	0.647	0.36	0.374	0.4	0.418	0.465	0.456	0.443	0.44	0.413	0.443
	192	0.398	0.414	0.413	0.422	0.422	0.431	0.396	0.421	0.672	0.653	0.381	0.371	0.434	0.445	0.722	0.588	0.471	0.455	0.459	0.465
	336	0.420	0.428	0.429	0.442	0.449	0.449	0.471	0.437	0.626	0.711	0.472	0.531	0.51	0.467	0.712	0.586	0.462	0.455	0.614	0.554
	720	0.440	0.460	0.456	0.474	0.53	0.513	0.454	0.463	0.835	0.797	0.49	0.488	0.636	0.544	0.581	0.533	0.525	0.503	0.45	0.464
	Average	0.406	0.424	0.417	0.433	0.447	0.451	0.422	0.43	0.7	0.702	0.426	0.441	0.495	0.469	0.62	0.541	0.475	0.463	0.484	0.482
ETTh1 → ETTm1	96	0.305	0.348	0.293	0.348	0.293	0.344	0.29	0.348	0.672	0.6	0.295	0.387	0.311	0.355	0.308	0.355	0.315	0.354	0.681	0.545
	192	0.342	0.369	0.326	0.368	0.327	0.366	0.327	0.372	0.721	0.639	0.335	0.379	0.337	0.372	0.357	0.39	0.365	0.391	0.689	0.551
	336	0.371	0.387	0.359	0.386	0.364	0.397	0.357	0.392	0.755	0.664	0.379	0.363	0.372	0.398	0.396	0.402	0.384	0.4	0.705	0.56
	720	0.424	0.416	0.411	0.418	0.409	0.417	0.409	0.423	0.837	0.705	0.403	0.431	0.422	0.433	0.419	0.423	0.428	0.426	0.722	0.571
	Average	0.361	0.380	0.347	0.380	0.348	0.381	0.346	0.384	0.746	0.652	0.353	0.39	0.36	0.39	0.37	0.393	0.373	0.393	0.699	0.557
Weather → ETTh1	96	0.365	0.392	0.371	0.395	0.386	0.409	0.477	0.444	-	-	-	-	0.397	0.44	0.421	0.41	0.428	0.429	0.393	0.41
	192	0.397	0.412	0.408	0.418	0.405	0.42	0.454	0.522	-	-	-	-	0.458	0.466	0.539	0.503	0.461	0.451	0.44	0.437
	336	0.423	0.429	0.427	0.438	0.448	0.454	0.424	0.434	-	-	-	-	0.479	0.458	0.568	0.514	0.463	0.456	0.45	0.451
	720	0.443	0.459	0.444	0.464	0.508	0.508	0.468	0.469	-	-	-	-	0.515	0.492	0.544	0.522	0.507	0.489	0.567	0.541
	Average	0.407	0.423	0.413	0.428	0.437	0.448	0.456	0.467	-	-	-	-	0.462	0.464	0.518	0.487	0.465	0.456	0.463	0.46
Weather → ETTm1	96	0.306	0.348	0.285	0.341	0.292	0.347	0.304	0.354	-	-	-	-	0.338	0.38	0.324	0.36	0.324	0.366	0.329	0.359
	192	0.336	0.365	0.327	0.370	0.332	0.373	0.338	0.375	-	-	-	-	0.473	0.457	0.359	0.387	0.349	0.377	0.392	0.392
	336	0.369	0.386	0.357	0.390	0.36	0.391	0.371	0.397	-	-	-	-	0.402	0.415	0.395	0.399	0.378	0.398	0.372	0.4
	720	0.424	0.416	0.419	0.420	0.406	0.421	0.417	0.426	-	-	-	-	0.432	0.438	0.45	0.467	0.422	0.427	0.434	0.429
	Average	0.359	0.379	0.347	0.380	0.348	0.383	0.358	0.388	-	-	-	-	0.411	0.423	0.382	0.403	0.368	0.392	0.382	0.395

effectiveness under cross-domain conditions. Both in-domain and cross-domain transfer settings, our model consistently achieves lower MSE and MAE than others, especially in ETTm1 → ETTh1, we outperform second best 4% in MSE, demonstrating strong generalization capability.

4.2.2 Time series Classification

In and Cross-domain Evaluation. For in-domain learning, We preform Epilepsy → Epilepsy following [18]. We adopt MLP as our encoder in this task can compare it with eight competitive SOTA baseline methods, including the contrastive learning (CL) based methods: TF-C, LaST, CoST, TS2Vec [20], and the masked time series modeling methods: SimMTM, Ti-MAE, TST [25]. As shown in table 4.6, Our model outperform second-best by 1.23% in F1. Similarly, on the challenging SleepEEG → FD-B transfer task, DeCoP achieves an average score of 91.83%, substantially higher than the second-best method, SimMTM, which

TABLE 4.6. For in-domain setting, we pre-train and fine-tune on the same dataset: Epilepsy. For cross-domain setting, we pre-train the model on SleepEEG and then fine-tune it on different datasets: Epilepsy, FD-B, and EMG. The best and the second best results are denoted by yellow and pink.

Scenarios		Models	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)	Avg (%)
In-Domain	Epilepsy → Epilepsy	TS2vec [20]	92.17	93.84	81.19	85.71	88.23
		CoST[35]	88.07	91.58	66.05	69.11	78.70
		LaST [52]	92.11	93.12	81.47	85.74	88.11
		TST [25]	80.21	40.11	50.00	44.51	53.71
		Ti-MAE [51]	90.09	93.90	77.24	78.21	84.86
		TF-C [53]	93.96	94.87	85.82	89.46	91.03
		SimMTM[18]	94.75	95.60	89.93	91.41	92.92
		DeCoP	95.32	92.64	92.69	92.64	93.32
Cross-Domain	SleepEEG → Epilepsy	TS2vec [20]	93.95	90.59	90.39	90.45	91.35
		CoST [35]	88.40	88.20	72.34	76.88	81.46
		LaST [52]	86.46	90.77	66.35	70.67	78.56
		TST [25]	80.21	40.11	50.00	44.51	53.71
		Ti-MAE [51]	89.71	72.36	67.47	68.55	74.52
		TF-C [53]	94.95	94.56	80.08	91.49	90.27
		SimMTM [18]	95.49	93.36	92.28	92.81	93.49
		DeCoP	95.73	93.17	93.40	93.29	93.90
	SleepEEG → FD-B	TS2Vec [20]	47.9	43.39	48.42	43.89	45.9
		CoST	47.06	38.79	38.42	34.79	39.77
		LaST [52]	46.67	43.9	47.71	45.17	45.86
		TST [25]	46.4	41.58	45.5	41.34	43.71
		Ti-MAE	60.88	66.98	68.94	66.56	65.84
		TF-C	69.38	75.59	72.02	74.87	72.97
		SimMTM	69.4	74.18	76.41	75.11	73.78
		DeCoP	89.73	92.65	92.48	92.47	91.83
	SleepEEG → EMG	TS2Vec [20]	78.54	80.4	67.85	67.66	71.97
		CoST	53.65	49.07	42.1	35.27	42.15
		LaST [52]	66.34	79.34	63.33	72.55	71.74
		TST [25]	78.34	77.11	80.3	68.89	75.43
		Ti-MAE [51]	69.99	70.25	63.44	70.89	68.19
		TF-C [53]	81.71	72.65	81.59	76.83	77.02
		SimMTM [18]	97.56	98.33	98.04	98.14	98.17
		DeCoP	97.56	98.33	98.04	98.14	98.17

achieves 73.78%. These results underscore the superiority of DeCoP in both in-domain and cross-domain scenarios, demonstrating its ability to handle complex classification tasks.

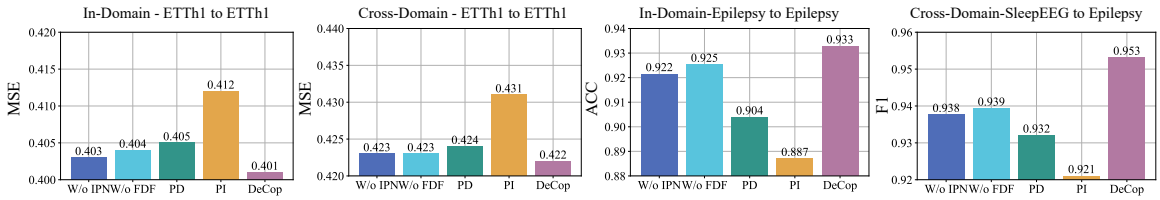

FIGURE 4.1. Ablation study of DeCoP, showing the impact of IPN, FDF, DCL, Patch Dependence (PD), and Patch Independence (PI) on time series forecasting (left) and classification (right) tasks in both in-domain and cross-domain settings.

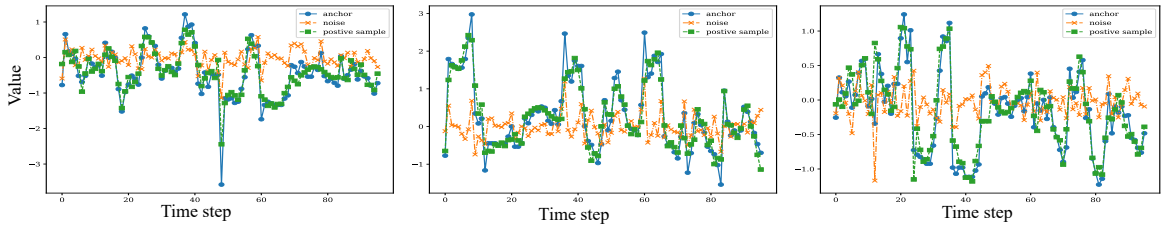

FIGURE 4.2. Generated positive sample pairs from the ETTh1 dataset with a sequence length of 100.

4.2.3 Ablation and analysis Study

Ablations. We conduct ablation studies on forecasting and classification tasks in both in- and cross-domain settings to assess the effectiveness of IPN and FDF with one without IPN and one without FDF. Additionally, we test two configurations for the DCL module: fully patch-independent (PI) and fully patch-dependent (PD). As shown in Figure 4.1, the performance drop 11% with PI and 0.4% with PD in forecasting task. Without IPN and FDF, the accuracy drop 1.55% and 1.38% individually in classification task.

Better Results with Reduced FLOPs. In this section, we compute the FLOPs and parameters of our framework compared to two SOTA pre-training frameworks, as shown in Table 4.7. In both pre-training and fine-tuning stages, DeCoP achieves the lowest MAE 0.421, significantly outperforming PatchTST by 24% on the ETTh1 dataset while using only 37% of the FLOPs.

Visualization of Positive Sample Based on FDF. To assess the effectiveness of FDF, we conducted ablation experiments on the parameter Top-k (filter intensity) as shown in Figure 3.4. Our model achieves lower MSE (0.401) when Top-$k = 0.1$ compared to no filter (0.406).

TABLE 4.7. Comparison of FLOPs, parameters, and average MAE on the ETTh1 dataset across different pre-training frameworks. Best result are denoted by yellow. Full results are in Appendix.

Models	Pretrain		Finetune		ETTh1
	FLOPs	Parameters	FLOPs	Parameters	MAE
PatchTST [22]	175.215M	598.028K	129.6M	**2.20M**	0.445
SimMTM [18]	4.269G	143.697M	99.893M	5.52M	0.428
DeCoP	**36.342M**	**479.244K**	**48.53M**	2.28M	**0.421**

MSE slightly increases to 0.404 at Top-$k = 0.5$, but still better than no filter. We also visualize the positive sample pairs generated by FDF in Figure 4.2, showing that filtered low-amplitude frequencies preserve the anchor samples' overall trend and cyclical patterns, ensuring semantic consistency.

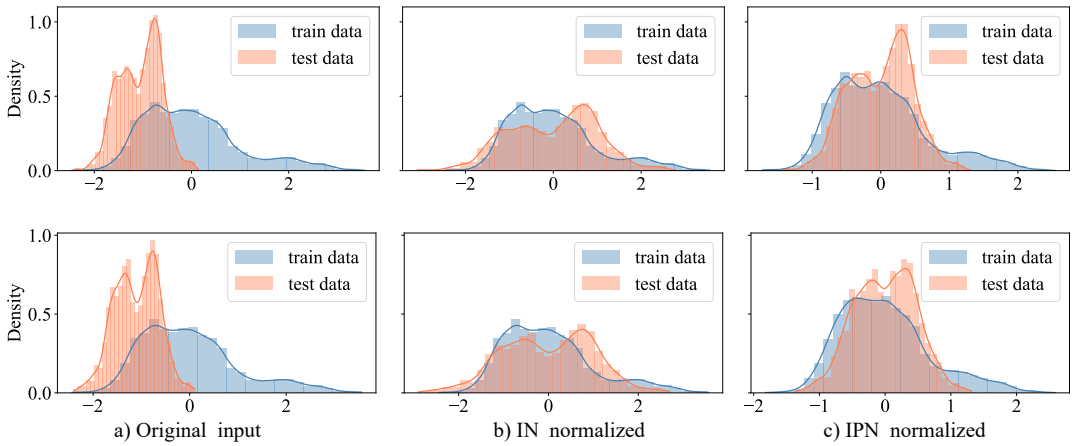

FIGURE 4.3. The data distribution for the ETTm1 and ETTh1 datasets is shown across three columns: (a), (b) and (c) represents the original data distribution, distribution after instance-wise normalization and after applying IPN individually.

The Effectiveness of IPN. We constructed a toy dataset with 99 samples that exhibit varying degrees of distribution shift between the training and test sets, as shown in Figure 3.2. All the differences are large than 0, which suggests that patch-wise normalization effectively enhances the model's ability to handle distribution shifts. We also compare the difference between Instance normalization and our IPN in Figure 4.3. Compared to IN, IPN maintains more consistent semantic variations with the original distribution.

TABLE 4.8. Effect of window sizes on dependency modeling. Optimal performance occurs with sizes around (2,5), while setting both windows to 1 or 42 causes a performance drop.

Window Size	ETTh1		ETTh2		ETTm1		ETTm2	
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
1,1	0.412	0.430	0.341	0.388	0.363	0.382	0.255	0.313
1,3	0.403	0.422	0.335	0.384	0.361	0.381	0.254	0.313
2,5	0.401	0.422	0.333	0.382	0.353	0.377	0.254	0.313
3,6	0.403	0.423	0.337	0.385	0.359	0.379	0.254	0.312
42,42	0.405	0.424	0.336	0.384	0.360	0.380	0.255	0.313

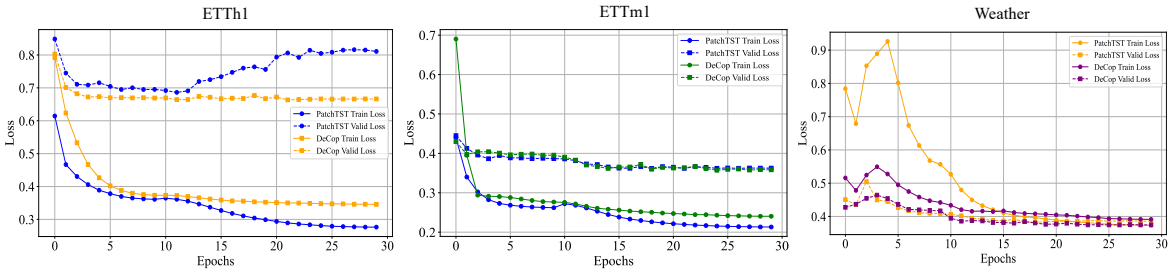

FIGURE 4.4. DeCoP is better at distribution shift than FDP framework. Here, we present the learning curves of PatchTST and DeCoP on three datasets highlight the differences in their performance. PatchTST exhibits overfitting issues, particularly on the ETTh1 dataset, while DeCoP effectively mitigates overfitting across all three datasets, demonstrating its robustness in handling diverse data distributions.

The robustness of DCL. The robustness of the Dependency-Controlled Learning (DCL) module is highly influenced by the selection of window sizes, as these sizes determine the extent of dependency modeling. A window size of 1 corresponds to fully independent learning, where each patch is treated in isolation, while a window size equal to the patch size enables fully dependent learning. Achieving an appropriate balance through optimal window sizes is crucial for stable and effective dependency modeling. To evaluate the impact of window sizes on DCL's performance, we tested various combinations—$(1, 1)$, $(1, 3)$, $(2, 5)$, $(3, 6)$, and $(42, 42)$—on four ETT datasets with forecast lengths of 96, 192, 336, and 720. The results are shown in Table 4.8. The model performs stably when the window size is set to 2 for layer 1 and 5 for layer 2, whereas fully dependent or fully independent modeling degrades performance by 0.8% and 0.4% individually.

TABLE 4.9. Comparison of DeCoP under different random seed across different datasets.

Datasets	Predicted length	DeCoP	
		MSE	MAE
Etth1	96	0.3613±0.0010	0.3903±0.0008
	192	0.3935±0.0010	0.4104±0.0007
	336	0.4143±0.0033	0.4275±0.0016
	720	0.4366±0.0015	0.4575±0.0009
Etth2	96	0.2679±0.0024	0.3329±0.0011
	192	0.3274±0.0019	0.3725±0.0034
	336	0.3533±0.0034	0.3974±0.0032
	720	0.3830±0.0024	0.4260±0.0022
Ettm1	96	0.2804±0.0033	0.3380±0.0014
	192	0.3232±0.0017	0.3648±0.0006
	336	0.3533±0.0038	0.3848±0.0010
	720	0.4089±0.0018	0.4180±0.0015
Ettm2	96	0.1631±0.0006	0.2538±0.0006
	192	0.2191±0.0004	0.2930±0.0002
	336	0.2682±0.0010	0.3263±0.0011
	720	0.3461±0.0008	0.3759±0.0010
Weather	96	0.1454±0.0005	0.1932±0.0007
	192	0.1896±0.0003	0.2367±0.0004
	336	0.2421±0.0001	0.2782±0.0008
	720	0.3141±0.0008	0.3289±0.0008
Electricity	96	0.1267±0.0001	0.2223±0.0001
	192	0.1457±0.0001	0.2359±0.0001
	336	0.1613±0.0003	0.2555±0.0004
	720	0.1950±0.0008	0.2890±0.0008

4.2.4 Comparison of Dependency-Controlled and Fully Dependent Pre-Training Frameworks

To assess the effectiveness of our proposed dependency controlled pre-training (DeCoP) framework, we compared the training and validation losses of DeCoP and PatchTST which relies on a fully dependent Pre-Training (FPD) framework, across three datasets over 30 training epochs, as illustrated in Figure 4.4. The results indicate that DeCoP effectively mitigates the distribution shift problem, as evidenced by its consistent decrease in both training and validation losses. In contrast, PatchTST exhibits signs of overfitting to the training dataset, likely due to its limited ability to handle distribution shifts effectively.

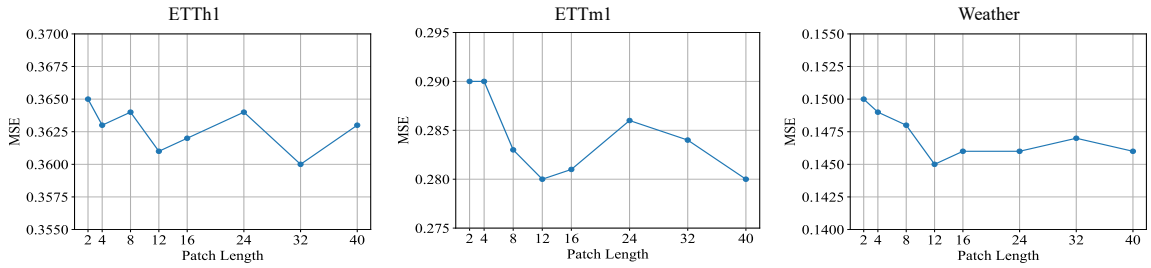

FIGURE 4.5. IPN Module Shows Robust Performance Across Patch Sizes. MSE scores are evaluated for patch lengths P = [2, 4, 8, 12, 16, 24, 32, 40] with a fixed look-back window of 512 and prediction length of 96. The results indicate small variation in MSE values, particularly for P = [8, 40], highlighting the IPN module's robustness.

4.2.5 Results with different random seeds

The results reported in the main text and appendix above are run with the fixed random seed 1. To examine the robustness of our results, we train the supervised PatchTST model with 5 different random seeds: 1,2,3,4,5 and calculate the MSE and MAE scores with each selected seed. The mean and standard derivation of the results are reported in Table 4.9. The findings demonstrate that the variances in MSE and MAE across different random seeds are notably small, indicating the stability and robustness of the model. This consistency suggests that DeCoP's performance is not significantly affected by random initialization, reinforcing the reliability of its predictions across different experimental setups. For instance, on the ETTh1 dataset with a prediction length of 96, the standard deviation of MSE is only ±0.0010, while the MAE deviation is ±0.0008. This trend is consistent across all datasets and prediction lengths, with standard deviations remaining below ±0.0040 in most cases, which validate that DeCoP's performance is not significantly influenced by random initialization, ensuring its robustness across multiple experimental setups. The low variability across random seeds highlights the model's ability to consistently capture temporal dependencies and maintain prediction accuracy, regardless of initialization settings. This robustness makes DeCoP a reliable choice for both research and real-world applications in time series forecasting.

4.2.6 The robustness of IPN module

This experiment explores the impact of varying patch lengths on the performance of the Instance-wise Patch Normalization (IPN) module. As previously mentioned, patch normalization is conducted along the patch size P, and here we examine the sensitivity of the IPN module to different patch lengths. The look-back window is fixed at 512, while the patch lengths are varied as P = [2, 4, 8, 12, 16, 24, 32, 40]. The stride length is set equal to the patch length, ensuring no overlap between patches. The model is trained to predict the next 96 time steps. As shown in Figure 4.5, the MSE scores remain relatively stable across different values of P, particularly within the range of 8 to 40, demonstrating the robustness of the IPN module. These observations suggest that a relatively large patch size is generally recommended to achieve optimal performance, depending on the dataset's characteristics. The consistent results across datasets demonstrate the ability of the IPN module to adapt to different configurations, thereby enhancing its robustness in handling varying patch lengths.

4.2.7 The generalizability of DCL strategy

As previously mentioned, we designed a Dependency Controlled Learning (DCL) module with a hierarchical controlled window. This experiment evaluates the impact of varying look-back lengths on the performance of DCL under fixed window sizes (2, 5). For this experiment, the patch length P and stride are fixed at 12, while the look-back window L = [96, 192, 336, 720] are varied. Additionally, we compare our model with the latest SOTA model, CycleNet, using different look-back lengths L = [96, 336, 720], where MSE and MAE are reported as in their paper. The result is shown in Table 4.10. As observed, the performance of DeCoP improves with an increase in look-back length, achieving the best performance across all three datasets.

4.2.8 The effectiveness of FDF method

To evaluate the robustness of the Frequency-domain Filter (FDF) method under varying Top$\mathcal{K}$ values, which control the intensity of noise filtering, we visualize the filtering process for

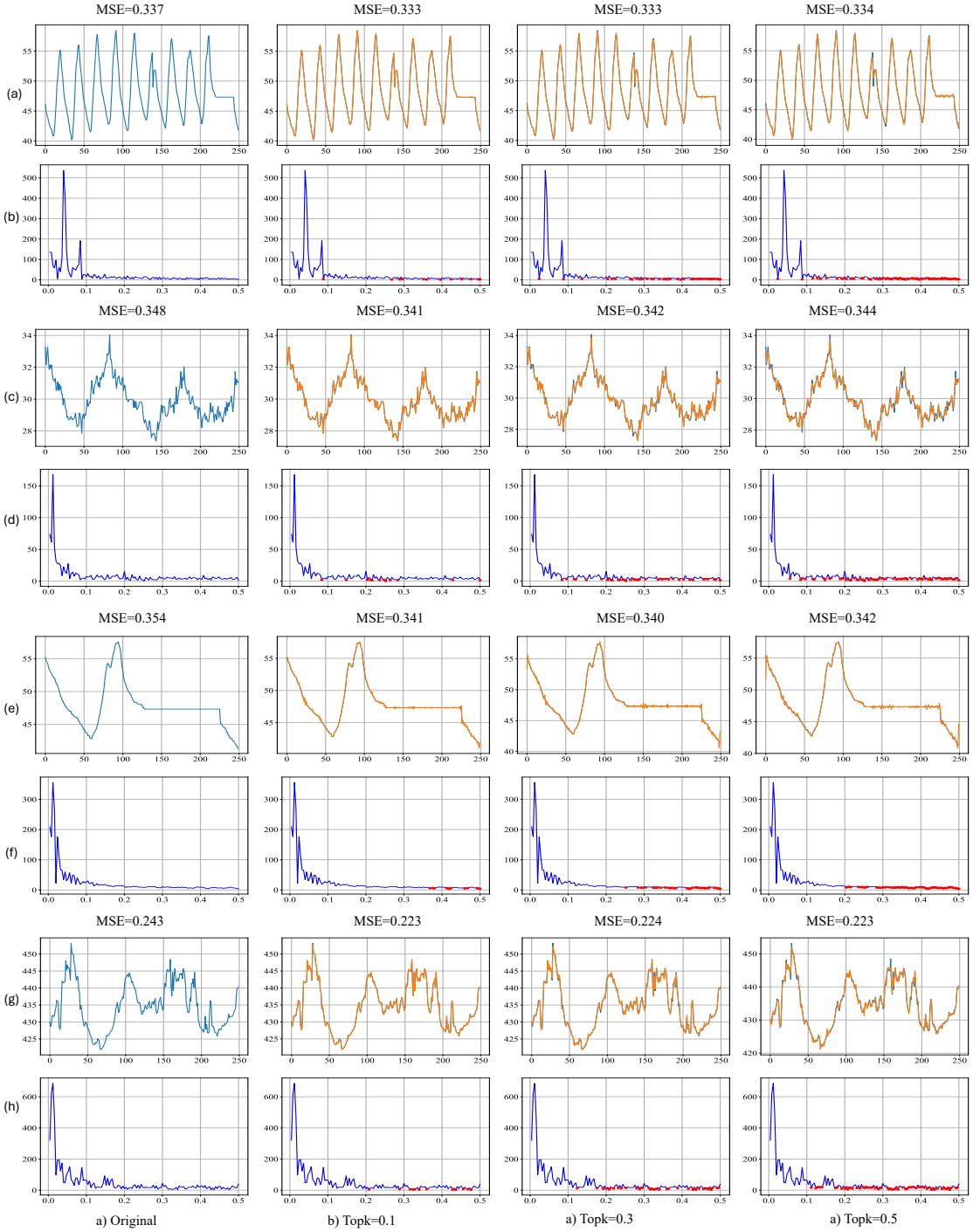

FIGURE 4.6. FDF is stable on different datasets with varying Top-k values. The first row displays the waveform, while the second row shows the amplitude spectrum of a randomly selected time series segment. Red dots indicate the filtered frequencies. (a-b) represent time series from ETTh2 dataset, (c-d) from ETTm1 dataset, (e-f) from ETTm2 dataset, and (g-h) from Weather dataset.

TABLE 4.10. Dependence-Controlled Learning Effectiveness Across Different Look-Back Times. This table compares the performance of DeCoP and CycleNet across various look-back windows and prediction horizons on three different datasets. The best result for each dataset is marked in yellow, and the best under each look-back time is marked in pink. DeCoP achieve allover best in each dataset and can get best result under the same look-back time once the input length larger than 96.

Dataset	Models	Look-back time	96		192		336		512		720	
			MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	DeCoP	96	0.384	0.394	0.380	0.396	0.370	0.393	0.361	0.390	0.363	0.394
		192	0.436	0.424	0.421	0.418	0.403	0.413	0.394	0.410	0.400	0.418
		336	0.478	0.446	0.453	0.436	0.428	0.429	0.414	0.427	0.426	0.436
		720	0.478	0.470	0.451	0.460	0.442	0.460	0.437	0.458	0.453	0.472
		Avg	0.444	0.433	0.426	0.428	0.411	0.424	0.401	0.421	0.411	0.430
	CycleNet	96	0.378	0.391	-	-	0.374	0.396	-	-	0.379	0.403
		192	0.426	0.419	-	-	0.406	0.415	-	-	0.416	0.425
		336	0.464	0.439	-	-	0.431	0.43	-	-	0.447	0.445
		720	0.461	0.46	-	-	0.45	0.464	-	-	0.477	0.483
		Avg	0.432	0.427	-	-	0.415	0.426	-	-	0.43	0.439
ETTm1	DeCoP	96	0.314	0.356	0.288	0.340	0.282	0.338	0.280	0.338	0.296	0.348
		192	0.358	0.380	0.323	0.363	0.324	0.365	0.323	0.365	0.332	0.369
		336	0.387	0.400	0.357	0.386	0.355	0.385	0.353	0.385	0.364	0.386
		720	0.449	0.435	0.417	0.421	0.411	0.418	0.409	0.418	0.412	0.415
		Avg	0.377	0.393	0.346	0.377	0.343	0.377	0.341	0.376	0.351	0.379
	CycleNet	96	0.319	0.36	-	-	0.299	0.348	-	-	0.307	0.353
		192	0.36	0.381	-	-	0.334	0.367	-	-	0.337	0.371
		336	0.389	0.403	-	-	0.368	0.386	-	-	0.364	0.387
		720	0.447	0.441	-	-	0.417	0.414	-	-	0.41	0.411
		Avg	0.379	0.396	-	-	0.355	0.379	-	-	0.355	0.381
Weather	DeCoP	96	0.175	0.216	0.157	0.201	0.148	0.195	0.145	0.193	0.144	0.192
		192	0.222	0.256	0.201	0.242	0.192	0.238	0.190	0.237	0.189	0.236
		336	0.277	0.296	0.255	0.283	0.244	0.278	0.242	0.278	0.242	0.279
		720	0.352	0.346	0.332	0.336	0.318	0.331	0.314	0.329	0.313	0.331
		Avg	0.256	0.278	0.236	0.266	0.226	0.261	0.223	0.259	0.222	0.260
	CycleNet	96	0.158	0.203	-	-	0.148	0.2	-	-	0.149	0.203
		192	0.207	0.247	-	-	0.19	0.24	-	-	0.192	0.244
		336	0.262	0.289	-	-	0.243	0.283	-	-	0.242	0.283
		720	0.344	0.344	-	-	0.322	0.339	-	-	0.312	0.333
		Avg	0.243	0.271	-	-	0.226	0.266	-	-	0.224	0.266

different Top$\mathcal{K}$ values: 0.0, 0.1, 0.3, and 0.5, as illustrated in Figure 4.6. The first row in each subfigure depicts the waveform of a randomly selected time series segment, while the second row shows its amplitude spectrum. Red dots highlight the frequencies filtered by FDF. As Top$\mathcal{K}$ increases, more low-amplitude frequencies are removed, while the critical high-amplitude components that define the primary characteristics of the time series remain intact.

TABLE 4.11. Full ablation studies ablation studies were conducted on in-domain learning tasks.

Scenarios	Models	w/o IPN		w/o FDF		PI		PD		DeCoP	
	Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1→ETTh1	96	0.362	0.392	0.363	0.392	0.371	0.399	0.361	0.390	0.361	0.390
	192	0.396	0.413	0.395	0.413	0.402	0.418	0.395	0.412	0.394	0.410
	336	0.417	0.427	0.417	0.428	0.425	0.434	0.418	0.428	0.414	0.427
	712	0.437	0.459	0.439	0.461	0.451	0.468	0.446	0.465	0.437	0.458
	Average	0.403	0.423	0.404	0.423	0.412	0.430	0.405	0.424	0.401	0.421
ETTh2→ETTh2	96	0.270	0.334	0.271	0.333	0.270	0.339	0.270	0.333	0.268	0.333
	192	0.342	0.379	0.341	0.377	0.337	0.380	0.336	0.384	0.327	0.373
	336	0.355	0.400	0.353	0.397	0.363	0.403	0.360	0.400	0.353	0.397
	712	0.386	0.430	0.382	0.426	0.391	0.430	0.388	0.426	0.383	0.426
	Average	0.338	0.386	0.337	0.383	0.341	0.388	0.336	0.384	0.333	0.382
ETTm1→ETTm1	96	0.282	0.341	0.287	0.345	0.307	0.353	0.284	0.339	0.280	0.338
	192	0.329	0.371	0.329	0.370	0.343	0.375	0.323	0.384	0.323	0.365
	336	0.358	0.385	0.361	0.389	0.370	0.386	0.355	0.389	0.353	0.385
	712	0.414	0.419	0.415	0.419	0.425	0.417	0.414	0.419	0.409	0.418
	Average	0.346	0.379	0.348	0.380	0.361	0.384	0.344	0.377	0.341	0.376
ETTm2→ETTm2	96	0.165	0.256	0.165	0.255	0.169	0.258	0.165	0.254	0.163	0.254
	192	0.217	0.292	0.219	0.292	0.228	0.298	0.219	0.293	0.210	0.293
	336	0.269	0.326	0.276	0.329	0.270	0.327	0.270	0.329	0.268	0.326
	712	0.354	0.380	0.357	0.382	0.377	0.395	0.381	0.395	0.346	0.376
	Average	0.251	0.314	0.254	0.316	0.262	0.314	0.252	0.314	0.249	0.312
Weather→Weather	96	0.149	0.197	0.171	0.225	0.154	0.204	0.145	0.196	0.145	0.193
	192	0.192	0.239	0.214	0.260	0.201	0.246	0.189	0.238	0.190	0.237
	336	0.243	0.278	0.260	0.295	0.248	0.282	0.240	0.279	0.242	0.278
	712	0.315	0.330	0.327	0.342	0.318	0.334	0.317	0.332	0.314	0.329
	Average	0.225	0.261	0.243	0.281	0.231	0.266	0.223	0.261	0.223	0.259
Electricity→Electricity	96	0.137	0.235	0.138	0.236	0.141	0.241	0.130	0.224	0.127	0.222
	192	0.153	0.249	0.153	0.249	0.155	0.252	0.148	0.240	0.146	0.236
	336	0.168	0.264	0.169	0.265	0.171	0.268	0.165	0.257	0.161	0.256
	712	0.208	0.298	0.210	0.308	0.214	0.306	0.196	0.299	0.195	0.289
	Average	0.166	0.261	0.167	0.264	0.170	0.267	0.160	0.255	0.157	0.251

This controlled filtering process ensures that the essential structure and dependencies within the data are preserved, enabling the model to focus on meaningful patterns while discarding irrelevant noise. In conclusion, the FDF method significantly enhances the performance of DeCoP by effectively mitigating noise and improving feature extraction across diverse datasets. The stability of the MSE across varying Top values demonstrates the robustness of the filtering mechanism, while the improvements observed for Top$\mathcal{K}$ >0 highlight its ability to enhance the model's generalization capabilities. These findings underscore the critical role of FDF in establishing a strong foundation for the subsequent dependence-controlled

TABLE 4.12. Full ablation studies ablation studies were conducted on cross-domain transfer tasks to ETTh1 and ETTm1 datasets. The experiments focus on forecasting future time points $F \in \{96, 192, 336, 720\}$ based on a look-back window of 512 past time points. Best results are denoted by yellow.

Scenarios	Models	w/o IPN		w/o FDF		PI		PD		DeCoP	
	Metric	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh2→ETTh1	96	0.364	0.394	0.362	0.392	0.376	0.404	0.361	0.391	0.361	0.391
	192	0.395	0.413	0.395	0.412	0.400	0.416	0.396	0.412	0.394	0.411
	336	0.417	0.427	0.417	0.428	0.424	0.433	0.418	0.428	0.419	0.427
	712	0.437	0.459	0.439	0.460	0.454	0.472	0.447	0.465	0.438	0.458
	Average	0.403	0.423	0.403	0.423	0.414	0.431	0.406	0.424	0.403	0.422
ETTm1→ETTh1	96	0.366	0.394	0.367	0.394	0.366	0.394	0.368	0.395	0.367	0.395
	192	0.397	0.413	0.399	0.414	0.403	0.416	0.399	0.413	0.398	0.414
	336	0.425	0.432	0.422	0.430	0.429	0.434	0.424	0.428	0.420	0.428
	712	0.442	0.462	0.444	0.463	0.452	0.467	0.447	0.463	0.440	0.460
	Average	0.408	0.425	0.408	0.426	0.412	0.428	0.410	0.425	0.406	0.424
ETTm2→ETTh1	96	0.367	0.394	0.368	0.395	0.365	0.393	0.369	0.395	0.362	0.391
	192	0.400	0.415	0.409	0.425	0.403	0.416	0.402	0.417	0.397	0.412
	336	0.427	0.433	0.433	0.439	0.429	0.435	0.427	0.432	0.421	0.429
	712	0.445	0.463	0.445	0.463	0.453	0.468	0.448	0.463	0.437	0.459
	Average	0.410	0.426	0.414	0.431	0.412	0.428	0.412	0.427	0.404	0.423
Weather→ETTh1	96	0.367	0.394	0.367	0.395	0.364	0.391	0.369	0.396	0.365	0.392
	192	0.402	0.417	0.425	0.439	0.398	0.412	0.399	0.413	0.397	0.412
	336	0.450	0.454	0.427	0.434	0.432	0.438	0.426	0.432	0.423	0.429
	712	0.450	0.468	0.450	0.468	0.442	0.461	0.447	0.464	0.443	0.459
	Average	0.417	0.433	0.417	0.434	0.409	0.425	0.410	0.426	0.407	0.423
ETTh2→ETTm1	96	0.285	0.341	0.283	0.340	0.304	0.352	0.294	0.347	0.283	0.340
	192	0.331	0.368	0.329	0.369	0.333	0.373	0.331	0.371	0.328	0.369
	336	0.359	0.387	0.361	0.388	0.365	0.393	0.360	0.390	0.354	0.388
	712	0.420	0.419	0.423	0.420	0.431	0.426	0.423	0.416	0.413	0.420
	Average	0.349	0.379	0.349	0.379	0.358	0.386	0.352	0.381	0.345	0.379
ETTh1→ETTm1	96	0.286	0.343	0.286	0.343	0.302	0.351	0.292	0.350	0.293	0.348
	192	0.329	0.369	0.331	0.371	0.332	0.372	0.326	0.367	0.326	0.368
	336	0.362	0.387	0.358	0.385	0.370	0.389	0.354	0.385	0.359	0.386
	712	0.417	0.419	0.418	0.418	0.429	0.419	0.418	0.419	0.411	0.418
	Average	0.349	0.379	0.348	0.379	0.358	0.383	0.348	0.380	0.347	0.380
ETTm2→ETTm1	96	0.284	0.341	0.285	0.342	0.301	0.350	0.288	0.341	0.284	0.340
	192	0.328	0.369	0.331	0.371	0.334	0.373	0.331	0.368	0.326	0.368
	336	0.360	0.387	0.362	0.386	0.375	0.391	0.358	0.387	0.357	0.387
	712	0.422	0.423	0.421	0.422	0.413	0.418	0.418	0.421	0.410	0.415
	Average	0.349	0.380	0.350	0.380	0.356	0.383	0.349	0.379	0.344	0.377
Weather→ETTm1	96	0.288	0.345	0.291	0.347	0.308	0.352	0.292	0.346	0.285	0.341
	192	0.335	0.368	0.333	0.375	0.339	0.375	0.334	0.371	0.327	0.370
	336	0.364	0.385	0.365	0.391	0.372	0.389	0.362	0.389	0.357	0.390
	712	0.423	0.417	0.424	0.423	0.419	0.419	0.421	0.414	0.419	0.420
	Average	0.353	0.379	0.353	0.384	0.359	0.384	0.352	0.380	0.347	0.380

TABLE 4.13. Ablation study conducted on classification tasks in both in-domain and cross-domain settings.

	Scenarios	Models	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)	Avg (%)
In-Domain	Epilepsy ↓ Epilepsy	w/o PN	95.00	93.37	90.53	91.86	92.69
		w/o FDF	95.19	93.47	91.09	92.22	92.99
		PI	92.04	88.83	85.24	86.87	88.24
		PN	93.00	91.80	85.36	88.09	89.56
		DeCoP	95.32	92.64	92.69	92.64	93.32
Cross-Domain	SleepEEG ↓ Epilepsy	w/o PN	95.20	94.60	89.94	92.04	92.94
		w/o FDF	95.72	93.64	92.75	93.19	93.82
		PI	90.68	85.59	84.73	85.15	86.54
		PN	94.23	93.93	87.42	90.21	91.45
		DeCoP	95.73	93.17	93.40	93.29	93.90
	SleepEEG ↓ FD-B	w/o PN	89.53	92.41	92.33	92.33	91.65
		w/o FDF	88.12	91.38	91.30	91.29	90.52
		PI	82.48	87.39	87.17	87.14	86.04
		PN	58.68	68.13	69.46	68.26	66.13
		DeCoP	89.73	92.65	92.48	92.47	91.83
	SleepEEG ↓ EMG	w/o PN	92.68	87.50	94.32	89.32	94.32
		w/o FDF	92.68	87.50	94.32	89.32	94.32
		PI	95.12	96.37	91.37	93.48	96.37
		PN	90.24	82.96	82.95	82.91	90.24
		DeCoP	97.56	98.33	98.04	98.14	98.33

learning stage, enabling DeCoP to handle complex temporal dependencies and achieve SOTA performance in forecasting tasks.

4.2.9 Full ablation study results

In this section, comprehensive ablation experiments are conducted on various forecasting and classification datasets under both in-domain and cross-domain settings. The experiments aim to evaluate the performance impact of different components in the proposed DeCoP framework and provide insights into the importance of dependency-controlled learning. For the forecasting task, in-domain learning results are presented in Table 4.11 while cross-domain transfer learning results are shown in Table 4.12. For the classification task, corresponding results are detailed in Table 4.13. Specifically, for the forecasting task, we perform ablation

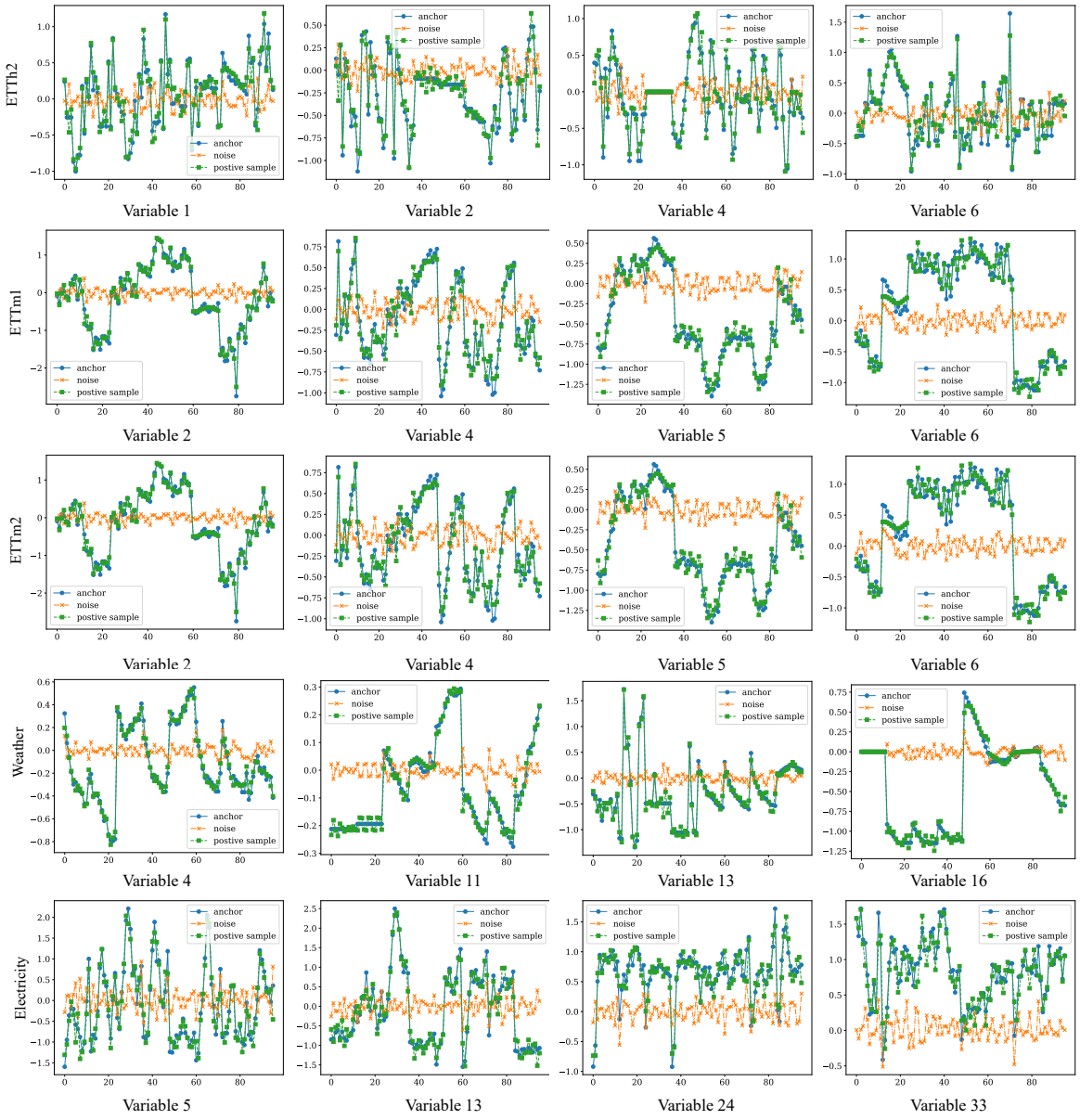

FIGURE 4.7. Visualization of positive sample pairs generated using FTF for forecasting datasets with a sequence length of 100. The variable index indicates channel order within each dataset. The blue line represents the anchor sample, while the green and orange lines depict the positive sample and filtered noise, respectively. The positive sample retains the anchor's primary characteristics with controlled variations in amplitude and temporal dynamics.

studies on DeCoP/Linear using datasets such as ETTh1 and ETTh2, where DeCoP achieves the best performance, consistently outperforming alternative configurations. Similarly, we evaluate DeCoP/MLP on datasets like ETTm1, ETTm2, Weather, and Electricity across

forecasting lengths ranging from 96 to 720. The experiments include four types of ablation designs: (1) without Instance-wise Patch Normalization (IPN), (2) without Fully Dependent Filtering (FDF), (3) Fully Patch Independent (PI), and (4) Fully Patch Dependent (PD). These experiments focus on understanding the impact of hierarchical dependency-controlled learning (DCL). The results (Table 4.11) demonstrate that DeCoP consistently outperforms all ablation settings across datasets and forecasting lengths. PI shows the weakest performance, highlighting the importance of modeling patch dependencies. Both IPN and FDF are critical components, as their removal results in a notable decline in performance. For example, in ETTh1 (Table 4.11), DeCoP achieves an average MSE of 0.401 compared to 0.412 with PI and 0.405 with PD. Similarly, on cross-domain tasks (Table 4.12), DeCoP maintains the best performance, with significant improvements observed in scenarios involving complex distribution shifts, such as ETTh2 $\rightarrow$ ETTm1 and Weather $\rightarrow$ ETTh1. For the classification task, DeCoP is tested on in-domain datasets such as Epilepsy and cross-domain datasets like SleepEEG $\rightarrow$ EMG. Results in Table 4.13 reveal that DeCoP achieves superior accuracy, precision, recall, and F1 scores across all scenarios. For instance, on the Epilepsy dataset, DeCoP achieves an average score of 93.32%, outperforming configurations without PN (92.69%) or FDF (92.99%). In cross-domain settings, such as SleepEEG $\rightarrow$ EMG, DeCoP achieves an outstanding average score of 98.33%, highlighting its robustness in learning transferable representations across domains.

4.2.10 More visualization results

In this section, we present additional examples of positive pairs generated by the FDF, as illustrated across six datasets in Figure 4.7.The generated positive samples effectively preserve the overall trends and cyclical patterns of the anchor samples by filtering out low-amplitude frequency components in the frequency domain. As we can see, compared to the anchor samples, the generated samples has similar fluctuations while more smooth by controlling the filter intensity, while the filtered noise primarily consists of random noise without any specific pattern. Utilizing these generated positive pairs with a contrastive learning process, DeCoP can learn better generalized high-level representations.

CHAPTER 5

Conclusion

This paper introduces **DeCoP**, a Dependency Controlled Pre-training framework that advances time series representation learning by balancing informative signals and noise. DeCoP addresses key challenges in time series modeling, including non-stationarity, sparse semantic context, and dependency management, by introducing two core components: Instance-wise Patch Normalization (IPN) and Hierarchical Dependency Controlled Learning (DCL). These components enable DeCoP to effectively reduce non-stationary noise, stabilize data distributions, and capture meaningful temporal dependencies across varying scales. Furthermore, the inclusion of a Frequency-domain Filtering (FDF) module enhances robustness by generating denoised positive samples, contributing to improved generalization across diverse time series patterns.

Extensive experiments on various dataset demonstrate the efficacy of DeCoP, achieving up to a 3% improvement in forecasting MSE compared to PatchTST in dataset ETTh1, while requiring only 37% of the computational cost. This significant reduction in resource requirements underscores the practical feasibility of DeCoP for real-world deployment in computationally constrained environments.

The findings of this study emphasize the importance of controlled dependency modeling in pre-training frameworks for time series tasks. DeCoP's ability to mitigate overfitting, disentangle temporal variations, and enhance representation clarity sets a strong foundation for further advancements in time series forecasting and classification. Additionally, by seamlessly integrating noise reduction and dependency stabilization, DeCoP establishes a pathway for future research to explore broader applications, including time series anomaly detection and time series imputation.

Bibliography

[1] H. Wu, T. Hu, Y. Liu, H. Zhou, J. Wang, and M. Long, "Timesnet: Temporal 2d-variation modeling for general time series analysis," *arXiv preprint arXiv:2210.02186*, 2022.

[2] A. Y. Barrera-Animas, L. O. Oyedele, M. Bilal, T. D. Akinosho, J. M. D. Delgado, and L. A. Akanbi, "Rainfall prediction: A comparative analysis of modern machine learning algorithms for time-series forecasting," *Machine Learning with Applications*, vol. 7, p. 100204, 2022.

[3] Y. Liu, H. Wu, J. Wang, and M. Long, "Non-stationary transformers: Exploring the stationarity in time series forecasting," *Advances in Neural Information Processing Systems*, vol. 35, pp. 9881–9893, 2022.

[4] A. Deng and B. Hooi, "Graph neural network-based anomaly detection in multivariate time series," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 35, pp. 4027–4035, 2021.

[5] X. Zhang, Z. Zhao, T. Tsiligkaridis, and M. Zitnik, "Self-supervised contrastive pre-training for time series via time-frequency consistency," *Advances in Neural Information Processing Systems*, vol. 35, pp. 3988–4003, 2022.

[6] H. Wu, H. Zhou, M. Long, and J. Wang, "Interpretable weather forecasting for worldwide stations with a unified deep model," *Nature Machine Intelligence*, vol. 5, no. 6, pp. 602–611, 2023.

[7] V. Ekambaram, K. Manglik, S. Mukherjee, S. S. K. Sajja, S. Dwivedi, and V. Raykar, "Attention based multi-modal new product sales time-series forecasting," in *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 3110–3118, 2020.

[8] M. Goswami, K. Szafer, A. Choudhry, Y. Cai, S. Li, and A. Dubrawski, "Moment: A family of open time-series foundation models," *arXiv preprint*, 2024.

[9] M. Jin, S. Wang, L. Ma, Z. Chu, J. Y. Zhang, X. Shi, and Q. Wen, "Time-llm: Time series forecasting by reprogramming large language models," *arXiv preprint*, 2023.

[10] Y. Liu, H. Zhang, C. Li, X. Huang, J. Wang, and M. Long, "Timer: Generative pre-trained transformers are large time series models," in *Proceedings of the Forty-first International Conference on Machine Learning*, 2024.

[11] G. Woo, C. Liu, A. Kumar, C. Xiong, S. Savarese, and D. Sahoo, "Unified training of universal time series forecasting transformers," in *Forty-first International Conference on Machine Learning*, 2024.

[12] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, B. McGrew, *et al.*, "Gpt-4 technical report," *arXiv preprint*, 2023.

[13] A. Dubey, A. Jauhri, A. Pandey, A. Kadian, A. Al-Dahle, A. Letman, A. Mathur, A. Schelten, A. Yang, A. Fan, *et al.*, "The llama 3 herd of models," *arXiv preprint arXiv:2407.21783*, 2024.

[14] B. Lim, S. Ö. Arık, N. Loeff, and T. Pfister, "Temporal fusion transformers for interpretable multi-horizon time series forecasting," *International Journal of Forecasting*, vol. 37, no. 4, pp. 1748–1764, 2021.

[15] T. Kim, J. Kim, Y. Tae, C. Park, J.-H. Choi, and J. Choo, "Reversible instance normalization for accurate time-series forecasting against distribution shift," in *International Conference on Learning Representations*, 2021.

[16] Y. Du, J. Wang, W. Feng, S. Pan, T. Qin, R. Xu, and C. Wang, "Adarnn: Adaptive learning and forecasting of time series," in *Proceedings of the 30th ACM international conference on information & knowledge management*, pp. 402–411, 2021.

[17] Z. Liu, J. Yang, M. Cheng, Y. Luo, and Z. Li, "Generative pretrained hierarchical transformer for time series forecasting," in *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 2003–2013, 2024.

[18] J. Dong, H. Wu, H. Zhang, L. Zhang, J. Wang, and M. Long, "Simmtm: A simple pre-training framework for masked time-series modeling," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[19] P. Xie, M. Ma, T. Li, S. Ji, S. Du, Z. Yu, and J. Zhang, "Spatio-temporal dynamic graph relation learning for urban metro flow prediction," *IEEE Transactions on Knowledge*

and Data Engineering, vol. 35, no. 10, pp. 9973–9984, 2023.

[20] Z. Yue, Y. Wang, J. Duan, T. Yang, C. Huang, Y. Tong, and B. Xu, "Ts2vec: Towards universal representation of time series," in *AAAI*, 2022.

[21] S. Lee, T. Park, and K. Lee, "Learning to embed time series patches independently," *arXiv preprint arXiv:2312.16427*, 2023.

[22] Y. Nie, N. H. Nguyen, P. Sinthong, and J. Kalagnanam, "A time series is worth 64 words: Long-term forecasting with transformers," *arXiv preprint arXiv:2211.14730*, 2022.

[23] B. Cai, S. Yang, L. Gao, and Y. Xiang, "Hybrid variational autoencoder for time series forecasting," *Knowledge-Based Systems*, vol. 281, p. 111079, 2023.

[24] Y. Zheng, H. Y. Koh, M. Jin, L. Chi, K. T. Phan, S. Pan, Y.-P. P. Chen, and W. Xiang, "Correlation-aware spatial–temporal graph learning for multivariate time-series anomaly detection," *IEEE Transactions on Neural Networks and Learning Systems*, 2023.

[25] G. Zerveas, S. Jayaraman, D. Patel, A. Bhamidipaty, and C. Eickhoff, "A transformer-based framework for multivariate time series representation learning," in *SIGKDD*, 2021.

[26] Z. Shao, Z. Zhang, F. Wang, and Y. Xu, "Pre-training enhanced spatial-temporal graph neural network for multivariate time series forecasting," in *Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining*, pp. 1567–1577, 2022.

[27] Z. Zhang, Y. Chen, D. Zhang, Y. Qian, and H. Wang, "Ctfnet: Long-sequence time-series forecasting based on convolution and time–frequency analysis," *IEEE Transactions on Neural Networks and Learning Systems*, 2023.

[28] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 2019.

[29] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, *et al.*, "Language models are few-shot learners," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[30] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *Proceedings of the 37th International Conference*

on Machine Learning, pp. 1597–1607, 2020.

[31] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *CVPR*, pp. 9729–9738, 2020.

[32] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *ICML*, 2020.

[33] T. Gao, X. Yao, and D. Chen, "Simcse: Simple contrastive learning of sentence embeddings," in *EMNLP*, 2021.

[34] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, *et al.*, "Learning transferable visual models from natural language supervision," in *International conference on machine learning*, pp. 8748–8763, PMLR, 2021.

[35] G. Woo, C. Liu, D. Sahoo, A. Kumar, and S. Hoi, "Cost: Contrastive learning of disentangled seasonal-trend representations for time series forecasting," in *ICLR*, 2022.

[36] X. Shi, S. Wang, Y. Nie, D. Li, Z. Ye, Q. Wen, and M. Jin, "Time-moe: Billion-scale time series foundation models with mixture of experts," *arXiv preprint arXiv:2409.16040*, 2024.

[37] H. Zhou, S. Zhang, J. Peng, S. Zhang, J. Li, H. Xiong, and W. Zhang, "Informer: Beyond efficient transformer for long sequence time-series forecasting," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 35, pp. 11106–11115, 2021.

[38] H. Wu, J. Xu, J. Wang, and M. Long, "Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting," *Advances in neural information processing systems*, vol. 34, pp. 22419–22430, 2021.

[39] T. Zhou, Z. Ma, Q. Wen, X. Wang, L. Sun, and R. Jin, "Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting," in *International conference on machine learning*, pp. 27268–27286, PMLR, 2022.

[40] A. Zeng, M. Chen, L. Zhang, and Q. Xu, "Are transformers effective for time series forecasting?," in *AAAI*, 2023.

[41] Y. Liu, T. Hu, H. Zhang, H. Wu, S. Wang, L. Ma, and M. Long, "itransformer: Inverted transformers are effective for time series forecasting," *arXiv preprint arXiv:2310.06625*, 2023.

[42] S. Lin, W. Lin, X. Hu, W. Wu, R. Mo, and H. Zhong, "Cyclenet: enhancing time series forecasting through modeling periodic patterns," *arXiv preprint arXiv:2409.18479*, 2024.

[43] E. Eldele, M. Ragab, Z. Chen, M. Wu, C. K. Kwoh, X. Li, and C. Guan, "Time-series representation learning via temporal and contextual contrasting," in *Proceedings of the 30th International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 2352–2359, International Joint Conferences on Artificial Intelligence Organization, 2021.

[44] Y. Özyurt, S. Feuerriegel, and C. Zhang, "Contrastive learning for unsupervised domain adaptation of time series," *CoRR*, vol. abs/2206.06243, 2022.

[45] Wetterstation, "Weather Dataset." https://www.bgc-jena.mpg.de/wetter/.

[46] UCI, "UCI Electricity Load Time Series Dataset." https://archive.ics.uci.edu/ml/datasets/ElectricityLoadDiagrams20112014.

[47] B. Kemp, A. H. Zwinderman, B. Tuk, H. A. Kamphuisen, and J. J. Oberye, "Analysis of a sleep-dependent neuronal feedback loop: the slow-wave microcontinuity of the eeg," *TBME*, 2000.

[48] R. G. Andrzejak, K. Lehnertz, F. Mormann, C. Rieke, P. David, and C. E. Elger, "Indications of nonlinear deterministic and finite-dimensional structures in time series of brain electrical activity: Dependence on recording region and brain state," *Physical Review E*, 2001.

[49] C. Lessmeier, J. K. Kimotho, D. Zimmer, and W. Sextro, "Condition monitoring of bearing damage in electromechanical drive systems by using motor current signals of electric motors: A benchmark data set for data-driven classification," in *PHM*, 2016.

[50] P. PhysioBank, "Physionet: components of a new research resource for complex physiologic signals," *Circulation*, 2000.

[51] Z. Li, Z. Rao, L. Pan, P. Wang, and Z. Xu, "Ti-mae: Self-supervised masked time series autoencoders," *arXiv preprint arXiv:2301.08871*, 2023.

[52] Z. Wang, X. Xu, W. Zhang, G. Trajcevski, T. Zhong, and F. Zhou, "Learning latent seasonal-trend representations for time series forecasting," in *NeurIPS*, 2022.

[53] X. Zhang, Z. Zhao, T. Tsiligkaridis, and M. Zitnik, "Self-supervised contrastive pre-training for time series via time-frequency consistency," in *NeurIPS*, 2022.

[54] S. Wang, H. Wu, X. Shi, T. Hu, H. Luo, L. Ma, J. Y. Zhang, and J. Zhou, "Timemixer: Decomposable multiscale mixing for time series forecasting," *arXiv preprint arXiv:2405.14616*, 2024.