# CoRA: Boosting Time Series Foundation Models for Multivariate Forecasting through Correlation-aware Adapter

**Anonymous authors**
Paper under double-blind review

## Abstract

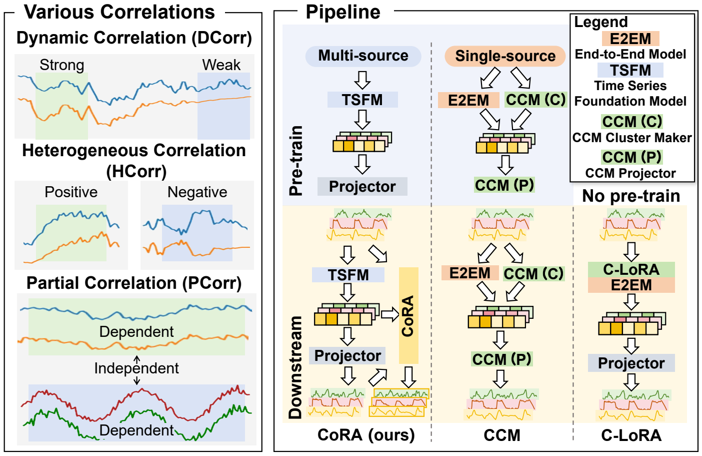
Most existing Time Series Foundation Models (TSFMs) use channel independent modeling and focus on capturing and generalizing temporal dependencies, while neglecting the correlations among channels or overlooking the different aspects of correlations. However, these correlations play a vital role in Multivariate time series forecasting. To address this, we propose a **CoR**relation-aware **A**dapter (**CoRA**), a lightweight plug-and-play method that requires only fine-tuning with TSFMs and is able to capture different types of correlations, so as to improve forecast performance. Specifically, to reduce complexity, we innovatively decompose the correlation matrix into low-rank Time-Varying and Time-Invariant components. For the Time-Varying component, we further design learnable polynomials to learn dynamic correlations by capturing trends or periodic patterns. To learn positive and negative correlations that appear only among some variables, we introduce a novel dual contrastive learning method that identifies correlations through projection layers, regulated by a Heterogeneous-Partial contrastive loss during training, without introducing additional complexity in the inference stage. Extensive experiments on 10 real-world datasets demonstrate that CoRA can improve the TSFMs in multivariate forecasting performance.

## 1 Introduction

Time Series Foundation Models (TSFMs) that show strong generalization are proposed recently. Through pre-training on large-scale time series data (Goswami et al., 2024; Liu et al., 2024e; Ekambaram et al., 2024a) or the use of large language models (Zhou et al., 2023; Liu et al., 2024d;c; Jin et al., 2023), these models maintain strong reasoning abilities when handling new or unseen data.

At the same time, multivariate time series forecasting, as a pivotal domain in data analysis, is widely applied in various industries (Qiu et al., 2024b; Wang et al., 2024b; Zhang et al., 2024). Properly modeling and utilizing correlations in multivariate time series can significantly improve the performance of forecasting models (Zhang & Yan, 2022; Liu et al., 2023; Wu et al., 2020). Based on different interaction characteristics among variables, as shown in Figure 1a, correlation can be summarized into three aspects: *dynamic correlation (DCorr)* describes the variation of variable relationships over time (Zhao et al., 2023; Cirstea et al., 2021); *heterogeneous correlation (HCorr)* focuses on how variables interact with each other by considering positive and negative correlations (Huang et al., 2023); *partial correlation (PCorr)* emphasizes that correlation exists only among certain variables, and modeling interactions across all variables can easily introduce noise (Chen et al., 2024; Qiu et al., 2025c; Liu et al., 2024b). Considering more comprehensive correlations provides richer information for the models.

However, most existing TSFMs focus on capturing and generalising temporal dependencies and neglect relationships among variables (Goswami et al., 2024; Ansari et al., 2024; Liu et al., 2024e;c; Jin et al., 2023; Shi et al., 2024). Although some models like TTM (Ekambaram et al., 2024a), UniTS (Gao et al., 2024), and Moirai (Woo et al., 2024) employ different methods to model the correlations among variables, they do not comprehensively consider multiple aspects of the correlations. For example, TTM employs an MLP-based channel mixing approach, but the MLP weights remain unchanged across different time steps, thereby failing to model DCorr while indiscriminately modelling all interactions, and thus failing to capture HCorr and PCorr explicitly. While the attention mechanisms used in UniTS and Moirai assign different attention scores at different time points, they still interact all variables simultaneously without considering HCorr and PCorr, thus leading to suboptimal correlation modeling. Furthermore, due to the variations in correlations across different datasets, it is difficult to capture generalized correlations during the pre-training phase (Ekambaram et al., 2024a).



(a) Various Correlations

(b) Efficient Plugins for Learning Correlations

Figure 1: (a) Illustration of three different types of correlations, the formal definitions are provided in Appendix A. (b) Comparisons of different plugins for learning correlations

Thus, it motivates us to design a plugin that can be fine-tuned alongside TSFMs, which avoids issues caused by correlation differences across datasets during the pre-training phase. Meanwhile, it possesses the ability to depict various correlations while also incorporating a lightweight design. However, this faces a major challenge: **balancing the complete modeling of various correlations with the lightweight design.** It is intrinsically difficult to model all three correlations in a unified manner. Although some models could address DCorr (Zhao et al., 2023; Cirstea et al., 2021), HCorr (Huang et al., 2023) and PCorr (Qiu et al., 2025c; Liu et al., 2024b) individually, they struggle to effectively encompass various correlations simultaneously. Moreover, existing variable interaction methods often rely on MLPs (Ekambaram et al., 2023; 2024b), Transformers (Liu et al., 2023; Jiang et al., 2023) and GNNs (Wu et al., 2020; Cai et al., 2024), etc., which have a time complexity of $\mathcal{O}(N^2)$, where $N$ denotes the number of variables. Some methods (Zhang & Yan, 2022; Chen et al., 2024; Nie et al., 2024) have made efforts in reducing the complexity. However, end-to-end models such as Crossformer (Zhang & Yan, 2022) require modifying or redesigning the entire model structure, and thus cannot be directly used as plugins for TSFMs. Existing plugins are primarily designed for end-to-end forecasting models. CCM (Chen et al., 2024) requires additional pre-training together with the end-to-end models before it can be plugged in. C-LoRA (Nie et al., 2024) is designed to be trained with an end-to-end backbone from scratch. Overall, there is a lack of an efficient plugin specifically designed for downstream fine-tuning of TSFMs. More importantly, considering various correlations in these methods would lead to a higher complexity.

To address this, we propose CoRA, a lightweight plug-and-play method that only requires training on a few samples with TSFMs during the fine-tuning phase. By considering various correlations, CoRA utilises representations and original prediction from TSFMs to generate an enhanced prediction, as shown in Figure 1b. To complete modeling the mentioned three types of correlation, we first propose the **Dynamic Correlation Estimation (DCE)** module which can learn dynamic correlation matrices. Then we design the **Heterogeneous-Partial Correlation Contrastive Learning (HPCL)** that uses the correlation matrices from DCE to learn HCorr and PCorr adaptively. Specifically, to achieve lightweight, we innovatively decompose the correlation matrices into two low-rank components: Time-Varying and Time-Invariant in DCE module. To better understand how DCorr evolves, we propose a learnable polynomial to capture trend or periodic patterns within the DCorr effectively. Afterwards, to better distinguish of HCorr, we propose channel-aware projections to map the representations into positive and negative correlation spaces. The projections are guided by the novel Heterogeneous-Partial Contrastive Loss during the training process, which enables adaptive learning of PCorr in the two HCorr spaces. As a result, we can capture the mentioned three types of correlations with $\mathcal{O}(N)$ complexity during inference. Our contributions are summarized as follows:

- We design a universal, lightweight plugin that allows the TSFMs to capture the mentioned three types of correlations without re-pre-training the TSFMs.
- We propose a lightweight Dynamic Correlation Estimation module that can explicitly model the dynamic patterns of correlations in a lightweight manner.
- We propose a novel Heterogeneous-Partial Correlation Contrastive Learning, which can learn HCorr and PCorr through projection layers regulated by dual contrastive loss.
- We conducted extensive experiments on 10 real-world datasets. The results show that CoRA effectively improves the performance of TSFMs in multivariate forecasting.

## 2 Related Work

### 2.1 Foundation Models for Time Series Forecasting

TSFMs for forecasting can be divided into two sections: **1) LLM-based Models:** These methods leverage the strong representational capacity and sequential modeling capability of LLMs to capture complex patterns for time series modeling. Among them, GPT4TS (Zhou et al., 2023) and CALF (Liu et al., 2024a) selectively modify certain parameters of LLMs to enable the model to adapt to time series data. On the other hand, UniTime(Liu et al., 2024c), $S^2$IP-LLM (Pan et al., 2024), LLMMixer (Kowsher et al., 2024), and Time-LLM (Jin et al., 2023) focus on creating prompts to trigger time series knowledge within LLMs. **2) Time Series Pre-trained Models:** Pre-training on multi-domain time series equips these models with strong generalization capabilities. Among them, ROSE (Wang et al., 2024a) and Moment (Goswami et al., 2024) restore the features of time series data, enabling them to extract valuable information in an unsupervised manner. On the other hand, TimesFM (Das et al., 2023) and Timer (Liu et al., 2024e), using an autoregressive approach, employ next-token prediction to learn time series representations. Generally speaking, most TSFMs are based on channel-independent strategies, with only a few (Gao et al., 2024; Ekambaram et al., 2024a; Woo et al., 2024) modeling relatively simple inter-variable relationships. The effects of more complex correlations in TSFMs remain under-explored.

### 2.2 Correlation of Variables in Time Series Forecasting

Channel correlation plays a crucial role in enhancing the predictions(Qiu et al., 2025a). They can be divided into specialized models and plugins from a paradigm perspective. **1) Correlation Models:** These models are typically based on foundational architectures such as MLP (Ekambaram et al., 2024b; 2023), GNN (Shang et al., 2021; Cai et al., 2024; Wu et al., 2020), and Transformer (Liu et al., 2023; Zhang & Yan, 2022). For example, TSMixer(Ekambaram et al., 2023) and TTM(Ekambaram et al., 2024b) directly mix all variables using MLP. MTGNN(Wu et al., 2020) and Ada-MsHyper(Shang et al., 2024) treat different variables as distinct nodes, performing message passing to facilitate variable interactions. Furthermore, iTransformer(Liu et al., 2023) and Crossformer(Zhang & Yan, 2022) treat different variables as distinct tokens and utilize transformers to realize channel interaction. **2) Correlation Plugins:** Some plugins enhance the predictive capability of models by learning correlation. For example, LIFT (Zhao & Shen, 2024) leverages locally stationary relationships to extract correlations. CCM (Chen et al., 2024) further performs clustering and creates dedicated prediction heads for each cluster. However, the methods above either lack comprehensive correlation modeling capabilities or possess substantial complexity.

## 3 Preliminaries

**Time Series Forecasting.** Given a multivariate time series with length $L$ and $N$ channels $\mathbf{X}_t = \{\mathbf{x}^i_{t-L:t}\}_{i=1}^N$, where each $\mathbf{x}^i_{t-L:t} \in \mathbb{R}^L$ is a sequence of observations at time point $t$. The forecasting task is to predict future $F$ length values $\hat{\mathbf{Y}}_t = \{\hat{\mathbf{x}}^i_{t:t+F}\}_{i=1}^N$. $\mathbf{Y}_t = \{\mathbf{x}^i_{t:t+F}\}_{i=1}^N$ denotes real future values.

**Correlation-Aware Adapter for Foundation Models.** Given a TSFM $\mathcal{F}$, it is fine-tuned on downstream forecasting data $\boldsymbol{X}_t^{ft}$, formulated as $\hat{\boldsymbol{Y}}_t^{ft} = \mathcal{F}(\boldsymbol{X}_t^{ft})$. Meanwhile, the series representation $\boldsymbol{\mathcal{X}}_t^{ft}$ of TSFM $\mathcal{F}$ is produced.

Problem Definition: given $\boldsymbol{X}_t^{ft}$, $\hat{\mathbf{Y}}_t^{ft}$ , $\boldsymbol{\mathcal{X}}_t^{ft}$ and $\mathbf{Y}_t^{ft}$, we update $\mathcal{F}$ into $\mathcal{F}^*$, where $\mathcal{F}^*$ is an updated $\mathcal{F}$ with CoRA plugged in. The inference can be performed as: $\hat{\mathbf{Y}}_t^{test} = \mathcal{F}^*(\boldsymbol{X}_t^{test})$.

# 4 Methodology

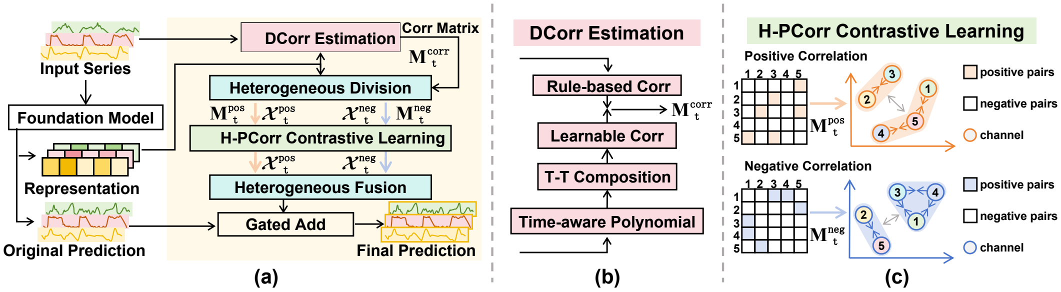


Figure 2: The framework of CoRA. (a) CoRA begins by learning DCorr in Dynamic Correlation Estimation module. Heterogeneous Division module projects representations into positive and negative spaces for HCorr. Then CoRA conducts H-PCorr Contrastive Learning in each space to guide projection and capture PCorr. (b) The DCorr Estimation module estimates correlations by combining Rule-based Correlations and Learnable Correlations, which are computed by Time-aware Polynomial and Time-Varying and Time-Invariant (T-T) Composition. (c) H-PCorr contrastive learning minimizes distances between strongly correlated channels and maximizes separation between weakly correlated channels in both positive and negative spaces.

In this work, we propose a **Cor**relation-Aware **A**dapter (CoRA), a lightweight plugin that allows the TSFMs to capture various correlations during the fine-tuning stage. The framework of CoRA is visualized in Figure 2 . CoRA operates on input series, original predictions, and representations from TSFMs to enhance the prediction accuracy. Our method consists of four processes: **(i) Dynamic Correlation Estimation.** This module utilize representations from TSFMs and input series to learn dynamic correlations and generate correlation matrices that guide subsequent contrastive learning. **(ii) Heterogeneous Division.** Some channels show dependencies on positive correlations, whereas some others show negative correlations. To better capture HCorr, we design the this module to process the representations from the backbone and learn representations of positive and negative correlations separately. **(iii) Heterogeneous Partial Correlation (H-PCorr) Contrastive Learning.** We propose H-PCorr Contrastive Learning within each representation of HCorr to learn PCorr by clustering only correlated channels. **(iv) Heterogeneous Fusion and prediction.** Finally, we fuse the representations after contrastive learning for positive and negative correlations in Heterogeneous Fusion module and generate new predictions. Then, both original and new predictions are gated and added together.

## 4.1 Dynamic Correlation Estimation

Channels exhibit both stable dependencies that do not change across time and fluctuations that change across time. Motivated by this, we introduce an innovative method that decomposes the learnable part of correlation matrix $\boldsymbol{M}_t^{corr} \in \mathbb{R}^{N\times N}$ at time $t$ into two low-rank components: Time-Varying $\boldsymbol{Q}_t \in \mathbb{R}^{N\times M}$ and Time-Invariant $\boldsymbol{V} \in \mathbb{R}^{M\times M}$, which can separate distinct correlation components, as illustrated in Figure 3. Here, $\boldsymbol{R} \in \mathbb{R}^{N\times N}$ denotes the rule-based correlation matrix which is added to the learnable part to incorporate more prior knowledge for enhancing correlation estimation. $M$ is the hyperparameter for the post-decomposition rank, with $M < N$. This decomposition of the learnable part offers greater parameter

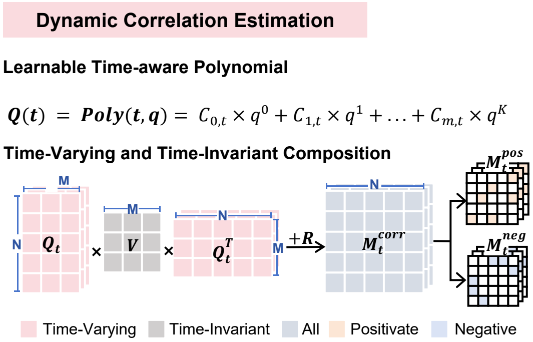

Dynamic Correlation Estimation
Learnable Time-aware Polynomial
$\boldsymbol{Q}(\boldsymbol{t}) = \boldsymbol{Poly}(\boldsymbol{t},\boldsymbol{q}) = C_{0,t} \times q^0 + C_{1,t} \times q^1 + \ldots + C_{m,t} \times q^K$
Time-Varying and Time-Invariant Composition
M
N
$Q_t$
×
M
$V$
×
N
$Q_t^T$
M
+R
N
$M_t^{corr}$
$M_t^{pos}$
$M_t^{neg}$
Time-Varying
Time-Invariant
All
Positivate
Negative

Figure 3: The details of DCorr Estimation

efficiency, yet remains functionally equivalent to conventional additive decomposition (Cirstea et al., 2021), as formally proven in Theorem 1.

We estimate the two components separately and then compose them back into the original correlation. The Time-Varying component represents the fluctuations in correlations across time. As time series data inherently have trends and periodic characteristics, the correlations that measure their dependencies also exhibit such variations across the entire time series (Liu et al., 2022). Thus, we propose Learnable Time-aware Polynomials to estimate the changes, as polynomials can be effective in modeling temporal patterns by sharing a common basis across different time steps. Based on a global adaptive method, the Time-Invariant component aims to capture the stable dependencies among channels that do not change over time. Finally, we compute the correlation matrix $\boldsymbol{M}_t^{corr}$ by composing learnable correlations and combining with the rule-based correlation $\boldsymbol{R}$. This correlation matrix is then used for H-PCorr Contrastive Learning.

### 4.1.1 Learnable Time-aware Polynomials

Most existing approaches (Shang et al., 2024; Cirstea et al., 2021; Zhao et al., 2023) struggle to accurately express the time-varying characteristics of DCorr due to the lack of explicit modeling of dynamic regularities.

In a stationary time series, we can use a well-behaved mathematical function to effectively approximate the fluctuations of the correlation. Considering that high-order polynomials provide better non-linear capacity than first-order ones, we use learnable polynomials to estimate $\boldsymbol{Q}_t$. The proof of this approximation capability is detailed in Theorem 2.

We construct a $K$-order Time-aware Polynomials with a shared matrix basis:

$$\boldsymbol{Q}_t = \sum_{i=0}^{K} C_{i,t}\boldsymbol{q}^i, \ (\boldsymbol{q}^i = \underbrace{\boldsymbol{q} \odot \boldsymbol{q} \odot \cdots \odot \boldsymbol{q}}_{i \text{ times}}) \ , \tag{1}$$

where $\boldsymbol{Q}_t \in \mathbb{R}^{N \times M}$ denote the Time-Varying component at time step t. $C_{i,t} \in \mathbb{R}^N$ is the $i$-th coefficient that varies over time, while $\boldsymbol{q} \in \mathbb{R}^{N \times M}$ is the globally learnable basis, which represents the pattern of changes over time. We define $\boldsymbol{q}^i$ as the i-times Hadamard product of the matrix $\boldsymbol{q}$, where the operation $\odot$ is the element-wise Hadamard product.

For convenience, we define the collection of $C_{i,t}$ as the matrix $\boldsymbol{\mathcal{C}}_t = (C_{0,t}, \cdots C_{K,t}) \in \mathbb{R}^{N \times K}$. It is the dependency coefficient of each channel for pattern $q^i$ and exhibits different values at different times, determined by specific data. Therefore, we learn the mapping $f$ between the representations of time series $\boldsymbol{\mathcal{X}}_t$ and coefficients $\boldsymbol{\mathcal{C}}_t$ to estimate it :

$$\boldsymbol{\mathcal{C}}_t = f(\boldsymbol{\mathcal{X}}_t) \in \mathbb{R}^{N \times K} \ . \tag{2}$$

Since only the polynomial coefficients need to be estimated with $f$, rather than the entire varying component, we can use a simple MLP to implement it.

### 4.1.2 Time-Varying and Time-Invariant Composition

Since the time-invariant part does not change over time, it should be globally unique. Inspired by self-learned graphs (Shang et al., 2024; Wu et al., 2020), we use global learnable vectors to capture the implicit stable dependencies among channels:

$$\boldsymbol{V} = \text{Sigmoid}(\text{ReLU}(\boldsymbol{E}_1\boldsymbol{E}_2^T)) \ , \tag{3}$$

where $\boldsymbol{V} \in \mathbb{R}^{M \times M}$ denote the Time-Invariant component of DCorr. $\boldsymbol{E}_1, \boldsymbol{E}_2 \in \mathbb{R}^{M \times d_e}$ are learnable vectors, $d_e$ is used to expand the dimensions, thereby enhancing the representation capacity.

As the statistics-based Pearson coefficient can describe simple linear correlation, we use it as the initialization for the final DCorr and build upon it to learn more complex correlations. The Pearson coefficient is calculated as follows:

$$r_{x,y} = \frac{\sum_{i=1}^{L}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{L}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{L}(y_i - \bar{y})^2}}, \tag{4}$$

where $x, y$ are two variables in $\boldsymbol{X}_t$. $r_{x,y}$ denotes the correlation coefficient among them, while $\bar{x}$ and $\bar{y}$ indicate the mean values of $x$ and $y$, respectively, $L$ is the size of input series. We use $\boldsymbol{R} \in \mathbb{R}^{N \times N}$ to denote the collection of $r$. Overall, our DCorr includes rule-based correlation $\boldsymbol{R}$, time-varying components $\boldsymbol{Q}_t$, and time-invariant components $\boldsymbol{V}$. The final estimated correlation is formulated as the sum of the learnable and rule-based parts in the following equation:

$$\boldsymbol{M}_t^{\text{corr}} = \boldsymbol{R} + \boldsymbol{Q}_t \boldsymbol{V} \boldsymbol{Q}_t{}^T. \tag{5}$$

### 4.2 Heterogeneous Correlation Division

Positive and negative correlations affect the channels differently, and the proportion of their contribution also varies among different channels. Motivated by Squeeze-and-Excitation (Hu et al., 2018) (SE), we propose a channel-aware projector that can adjust the weights of channels based on contextual information during projection to better project the representations into positive and negative spaces and learn Heterogeneous Correlation (HCorr).

To distinguish the dependency of channels on Heterogeneous Correlation, we project the representations into two spaces. Specifically, we use SE mechanism to aggregate contextual information over time and adaptively calculate projection weights among channels. The channel-aware projection layer $\mathcal{P}$ with $\boldsymbol{\mathcal{X}}_t^{\text{in}}$ and $\boldsymbol{\mathcal{X}}_t^{\text{out}}$ is shown as follows:

$$\boldsymbol{\mathcal{X}}_t^{\text{proj}} = \text{MLP}_1(\text{LayerNorm}(\boldsymbol{\mathcal{X}}_t^{\text{in}})) \in \mathbb{R}^{P \times N \times d}\,, \tag{6}$$

$$W = \text{SoftMax}(\text{MLP}_2(\text{LayerNorm}(\boldsymbol{\mathcal{X}}_t^{\text{in}}))) \in \mathbb{R}^N\,, \tag{7}$$

$$\boldsymbol{\mathcal{X}}_t^{\text{out}} = \boldsymbol{\mathcal{X}}_t^{\text{in}} + \boldsymbol{\mathcal{X}}_t^{\text{proj}} \odot \text{expand}(W)\,, \tag{8}$$

where LayerNorm refers to the layer normalization operation, which enhances the model's generalisation ability. $\text{MLP}_1 := \mathbb{R}^d \to \mathbb{R}^d$ is used for preliminary projection. $\text{MLP}_2 := \mathbb{R}^{P \times d} \to \mathbb{R}^1$ is used to compute channel weights. W denotes the adaptive channel weight.

We perform two identical projection transformations on $\boldsymbol{\mathcal{X}}_t$ to obtain the representations of the positive and negative latent spaces:

$$\boldsymbol{\mathcal{X}}_t^{\text{pos}} = \mathcal{P}_1(\boldsymbol{\mathcal{X}}_t) \in \mathbb{R}^{(P \times N \times d)},\ \boldsymbol{\mathcal{X}}_t^{\text{neg}} = \mathcal{P}_2(\boldsymbol{\mathcal{X}}_t) \in \mathbb{R}^{(P \times N \times d)}\,, \tag{9}$$

where $\mathcal{P}_1$ and $\mathcal{P}_2$ are the same channel-aware projection operations, and they consist of $N_1$ projection layers like $\mathcal{P}$. $\boldsymbol{\mathcal{X}}_t^{\text{pos}}$ and $\boldsymbol{\mathcal{X}}_t^{\text{neg}}$ are representations projected into spaces of positive and negative correlations, respectively. They contain channel information with adaptive adjustments and will subsequently be used for contrastive learning.

It is noteworthy that the Heterogeneous Correlation Division module cannot directly accomplish the disentanglement of heterogeneous correlations. Instead, this separation is achieved under the guidance of the contrastive learning framework detailed in the next section.

### 4.3 Heterogeneous Partial Correlation Contrastive Learning

To capture Partial Correlation, we design Partial Contrastive Learning, which uses the correlation matrix derived from Dynamic Correlation Estimation (DCE) and representations from Heterogeneous Correlation Division (HD) to enable adaptive cluster learning.

We leverage Contrastive Learning's advantages for clustering to capture PCorr. Compared to existing methods (Chen et al., 2024; Qiu et al., 2025c), this approach facilitates the fine-grained interaction among relevant channels. Moreover, it does not add an extra burden during inference.

First based on the estimated correlation $\boldsymbol{M}_t^{corr}$, we define the heterogeneous correlations as $\boldsymbol{M}_t^{\text{pos}}$ and $\boldsymbol{M}_t^{\text{neg}}$ to decouple the complex interactions among variables:

$$\boldsymbol{M}_t^{\text{pos}} = \begin{cases} m_t^{\text{corr}}, & \text{if } corr > \epsilon \\ 0, & else \end{cases},\ \boldsymbol{M}_t^{\text{neg}} = \begin{cases} m_t^{\text{corr}}, & \text{if } corr < -\epsilon \\ 0, & else \end{cases}\,, \tag{10}$$

where $m_t^{\text{corr}}$ is the element of $\boldsymbol{M}_t^{\text{corr}}$, $\epsilon$ is the learnable threshold. The following process is for the positive correlation in the positive latent space; the same operation is performed on the negative latent spaces.

The matrix $\boldsymbol{M}_t^{\text{pos}}$ is used to select positive and negative samples for each variable. In the designed contrastive learning if $\boldsymbol{M}_t^{\text{pos}}[i,j] = 0$, it is considered a negative pair; otherwise, it is considered a positive pair. The loss for the positive correlation can be expressed as follows:

$$\mathcal{L}_{pos} = -\frac{1}{N}\sum_{i=1}^{N} log(\frac{\sum_{j=1}^{N} \boldsymbol{M}_t^{\text{pos}}[i,j]\text{exp}(\text{sim}(\boldsymbol{\mathcal{X}}_t^{\text{pos}}[i], \boldsymbol{\mathcal{X}}_t^{\text{pos}}[j])/\tau)}{\sum_{k=1}^{N} \text{exp}(\text{sim}(\boldsymbol{\mathcal{X}}_t^{\text{pos}}[i], \boldsymbol{\mathcal{X}}_t^{\text{pos}}[k])/\tau)}), \tag{11}$$

where $\text{sim}(\cdot)$ represents the cosine similarity, and $\tau$ is the temperature coefficient used to control the degree of contrastive learning constraints. The following equation gives the final loss:

$$\mathcal{L} = \gamma(\mathcal{L}_{pos} + \mathcal{L}_{neg}) + \mathcal{L}_{Forecast}, \tag{12}$$

where $\gamma$ is the tuning coefficient, and $\mathcal{L}_{Forecast}$ is the forecasting loss.

### 4.4 Heterogeneous Fusion and prediction

Finally, we project the representations of the two heterogeneous latent spaces into a shared space and then fuse them to perform prediction. Considering that some channels may require more correlation interaction while others may require more independence, we conduct a convex combination:

$$\tilde{\boldsymbol{\mathcal{X}}}_t^{\text{pos}} = \mathcal{P}_3(\boldsymbol{\mathcal{X}}_t^{\text{pos}}),\ \tilde{\boldsymbol{\mathcal{X}}}_t^{\text{neg}} = \mathcal{P}_4(\boldsymbol{\mathcal{X}}_t^{\text{neg}}), \tag{13}$$

$$\hat{\boldsymbol{Y}}_t^{*} = \beta\, \text{Linear}(\tilde{\boldsymbol{\mathcal{X}}}_t^{\text{pos}} + \tilde{\boldsymbol{\mathcal{X}}}_t^{\text{neg}}) + (1-\beta)\, \hat{\boldsymbol{Y}}_t\ , \tag{14}$$

where $\mathcal{P}_3$ and $\mathcal{P}_4$ consist of $N_2$ projection layers like $\mathcal{P}$, as given by equations (6-8). Linear represents the linear prediction head, $\beta \in [0,1]^N$ is a learning parameter denotes the gated weight.

### 4.5 Complexity Analysis

The computational complexities are $\mathcal{O}(N^2)$ for the DCorr Estimation (DCE, Section 4.1) and H-PCorr Contrastive learning (HPCL, Section 4.3), and $\mathcal{O}(N)$ for HCorr Division (HD, Section 4.2). Most of the complexity arises from DCE and HPCL, which are only required during training. In the inference phase, since CoRA only includes HD modules, the time complexity is $\mathcal{O}(N)$.

Figure 5 shows CoRA imposes only minimal additional time on TSFMs, during fine-tuning and inference. The details of complexity analysis are included in Appendix B.

## 5 Theoretical Analysis

### 5.1 The significance of Time-Varying and Time-Invariant Composition

A straightforward approach to modeling dynamic correlations is to decompose the correlation matrix into the sum of a time-varying matrix and a time-invariant matrix (Cirstea et al., 2021; Wu et al., 2019). However, this approach has parameter complexity. Our method can reduce the complexity while achieving the same effect.

**Theorem 1** *When the time series is locally stationary, the Time-Varying and Time-Invariant Decomposition allows $\boldsymbol{Q}_t\boldsymbol{V}\boldsymbol{Q}_t^T$ to contain both time-varying and time-invariant information, like conventional additive decomposition.*

*Specifically, $\boldsymbol{Q}_t\boldsymbol{V}\boldsymbol{Q}_t^T$ can be expressed as the sum of a time-invariant matrix $\boldsymbol{M}_i$ and a time-varying matrix $\boldsymbol{M}_v$, as shown below:*

$$\boldsymbol{Q}_t\boldsymbol{V}\boldsymbol{Q}_t^T = \boldsymbol{M}_i + \boldsymbol{M}_v\ . \tag{15}$$

This indicates that our decomposition approach remains functionally equivalent to conventional additive decomposition. Notably, the expression $\boldsymbol{Q}_t\boldsymbol{V}\boldsymbol{Q}_t^T$ is the learnable component of the correlation matrix $\boldsymbol{M}_t^{corr}$, as defined in Equation 5.

### 5.2 The fitting ability of Time-aware Polynomials

Time-aware polynomials can model complex time-varying correlation relationships, and the error bound decreases as the degree $K$ of the polynomial increases.

**Theorem 2** *When the time series is locally stationary, we can approximate the underlying correlation matrix with a high-order polynomial.*

*Specifically, assuming that the correlation is a smooth function of the basis* $\boldsymbol{q}$*, the true correlation component* $\boldsymbol{Q}_t^*$ *can be expressed as* $\mathcal{F}(\boldsymbol{q})$*. The error bound can be formalized as follows.*

$$|\boldsymbol{Q}_t^* - \boldsymbol{Q}_t| = \frac{\mathcal{F}^{(K+1)}(\boldsymbol{\xi})}{(K+1)!}\boldsymbol{q}^{(K+1)}, \boldsymbol{\xi} \in [-|\boldsymbol{q}|, |\boldsymbol{q}|] \,. \tag{16}$$

This indicates that by selecting an appropriate $K$, we can strike an effective balance between model effectiveness and computational efficiency, thereby enabling the efficient estimation of dynamic correlations. We provide the proof of Theorems 1-2 in the Appendix C.

## 6 Experiment

### 6.1 Experimental Details

**Datasets.** To conduct comprehensive and fair comparisons for different models, we conduct experiments on ten well-known forecasting benchmarks as the target datasets, including ETT (4 subsets), Electricity, Traffic, Solar, weather, AQShunyi and ZafNoo, which cover multiple domains. More details of the benchmark datasets are included in Table 4 of Appendix D.1.

**Baselines and Implementation.** We choose the latest state-of-the-art models to serve as baselines, including 3 Time Series LLM-based models (GPT4TS, AutoTimes, UniTime) and 3 Time Series pre-trained models (Moment, Chronos, Timer). We utilize the FM4TS-Bench Li et al. (2025) code repository for unified evaluation. More implementation details are included in D.3. To keep consistent with previous works, we adopt Mean Squared Error (MSE) and Mean Absolute Error (MAE) as evaluation metrics. We provide our code at https://anonymous.4open.science/r/CoRA-D968.

### 6.2 Main Results

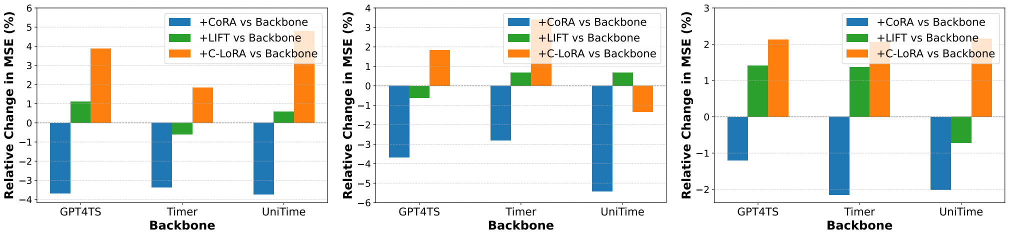
Comprehensive forecasting results of TSFMs with and without using CoRA are listed in Table 1. We have the following observations: i) Compared to fine-tuning without CoRA, fine-tuning with CoRA achieves better results in average results and results of different forecasting horizons (Table 6 and Table 7, in Appendix E) for both LLM-based models and time series pre-trained models, even in 5% few-shot settings. ii) Sharing the same pre-trained parameters, TTM's Channel-Dependent (CD) and Channel-Independent (CI) versions differ only in their module configurations during downstream fine-tuning. We implement a CI version of TTM, fine-tuned with CoRA and compare it with a CD version which is fine-tuned without CoRA. The better performance of the former demonstrates that considering the mentioned three types of correlations allows the model to better understand the inter-channels interaction.

### 6.3 Comparison with Other Correlation Plugins

To better validate the effectiveness of CoRA, we compare it with LIFT Zhao & Shen (2024) and C-LoRA Nie et al. (2024). We select GPT4TS, UniTime and Timer as the backbone and set H to 96. As shown in Figure 4, since LIFT and C-LoRA are not specifically designed for TSFMs, the limited training samples in the few-shot setting lead to a degradation in their performance, negatively impacting the effectiveness of the TSFMs. In contrast, CoRA, designed specifically for TSFMs, learns multiple correlations from the TSFMs' representations, allowing it to fully leverage their predictive capabilities. More comparisons with other fine-tuning setting are included in the Appendix F.1.

Table 1: Multivariate forecasting results in the 5% few-shot setting with MSE are averaged across four different forecasting horizons $H \in \{96, 192, 336, 720\}$.The better results are highlighted in **bold**. Full and MAE results are available in Appendix E.

| Model | LLM-Based | | | | | | Pre-trained | | | | | | Confidence Interval |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | GPT4TS (2023) | | CALF (2025) | | UniTime (2024) | | Moment (2024) | | Timer (2024) | | TTM (2024) | | |
| Plugin | ✗ | ✓ | ✗ | ✓ | ✗ | ✓ | ✗ | ✓ | ✗ | ✓ | ✗ | ✓ | |
| ETTh1 | 0.468<br>±0.002 | **0.456**<br>**±0.001** | 0.444<br>±0.003 | **0.433**<br>**±0.001** | 0.739<br>±0.002 | **0.712**<br>**±0.001** | 0.551<br>±0.003 | **0.537**<br>**±0.002** | 0.446<br>±0.003 | **0.432**<br>**±0.001** | 0.406<br>±0.002 | **0.393**<br>**±0.001** | 99% |
| ETTh2 | 0.377<br>±0.003 | **0.361**<br>**±0.002** | 0.376<br>±0.003 | **0.365**<br>**±0.001** | 0.399<br>±0.003 | **0.385**<br>**±0.001** | 0.369<br>±0.003 | **0.356**<br>**±0.002** | 0.357<br>±0.003 | **0.343**<br>**±0.001** | 0.345<br>±0.002 | **0.331**<br>**±0.002** | 99% |
| ETTm1 | 0.390<br>±0.003 | **0.378**<br>**±0.002** | 0.375<br>±0.003 | **0.363**<br>**±0.001** | 0.407<br>±0.002 | **0.392**<br>**±0.002** | 0.455<br>±0.003 | **0.439**<br>**±0.001** | 0.359<br>±0.003 | **0.346**<br>**±0.001** | 0.358<br>±0.002 | **0.344**<br>**±0.001** | 95% |
| ETTm2 | 0.279<br>±0.002 | **0.267**<br>**±0.001** | 0.274<br>±0.003 | **0.263**<br>**±0.002** | 0.293<br>±0.003 | **0.278**<br>**±0.002** | 0.277<br>±0.002 | **0.270**<br>**±0.002** | 0.262<br>±0.003 | **0.250**<br>**±0.001** | 0.259<br>±0.003 | **0.249**<br>**±0.002** | 99% |
| Electricity | 0.207<br>±0.003 | **0.201**<br>**±0.001** | 0.175<br>±0.003 | **0.166**<br>**±0.001** | 0.202<br>±0.003 | **0.191**<br>**±0.001** | 0.200<br>±0.003 | **0.196**<br>**±0.001** | 0.242<br>±0.002 | **0.229**<br>**±0.002** | 0.181<br>±0.002 | **0.173**<br>**±0.001** | 99% |
| Traffic | 0.441<br>±0.001 | **0.430**<br>**±0.001** | 0.435<br>±0.002 | **0.424**<br>**±0.001** | 0.456<br>±0.002 | **0.444**<br>**±0.001** | 0.453<br>±0.003 | **0.437**<br>**±0.002** | 0.458<br>±0.003 | **0.439**<br>**±0.001** | 0.486<br>±0.004 | **0.468**<br>**±0.001** | 95% |
| Solar | 0.254<br>±0.003 | **0.244**<br>**±0.001** | 0.229<br>±0.004 | **0.223**<br>**±0.001** | 0.252<br>±0.005 | **0.245**<br>**±0.002** | 0.226<br>±0.003 | **0.218**<br>**±0.002** | 0.217<br>±0.003 | **0.207**<br>**±0.002** | 0.269<br>±0.004 | **0.259**<br>**±0.001** | 99% |
| Weather | 0.254<br>±0.003 | **0.243**<br>**±0.001** | 0.238<br>±0.002 | **0.229**<br>**±0.002** | 0.255<br>±0.002 | **0.240**<br>**±0.002** | 0.251<br>±0.003 | **0.243**<br>**±0.001** | 0.247<br>±0.004 | **0.238**<br>**±0.002** | 0.226<br>±0.002 | **0.214**<br>**±0.002** | 99% |
| AQShunyi | 0.849<br>±0.002 | **0.830**<br>**±0.002** | 0.732<br>±0.003 | **0.714**<br>**±0.001** | 0.743<br>±0.003 | **0.715**<br>**±0.001** | 0.693<br>±0.003 | **0.670**<br>**±0.002** | 0.736<br>±0.004 | **0.708**<br>**±0.001** | 0.701<br>±0.003 | **0.678**<br>**±0.002** | 99% |
| ZafNoo | 0.564<br>±0.003 | **0.552**<br>**±0.002** | 0.549<br>±0.002 | **0.532**<br>**±0.002** | 0.563<br>±0.003 | **0.540**<br>**±0.001** | 0.533<br>±0.003 | **0.516**<br>**±0.002** | 0.539<br>±0.002 | **0.517**<br>**±0.002** | 0.505<br>±0.003 | **0.483**<br>**±0.001** | 99% |



(a) ETTm2 (b) Weather (c) Electricity

Figure 4: Correlation Plugins comparison in 5% few-shot fine-tuning setting.

### 6.4 Ablation Study

To investigate the effectiveness of CoRA, we conduct comprehensive experiments. In our work, the DCorr Estimation (DCE) is used to learn DCorr and generate the labels required for H-PCorr Contrastive Learning (HPCL), while HPCL utilizes these labels to guide the projectors in the Heterogeneous Division (HD) module. Therefore, they cannot operate independently. We utilize naive implementations in place of the original modules within certain variants.

Table 2: The MSE results of various variants.

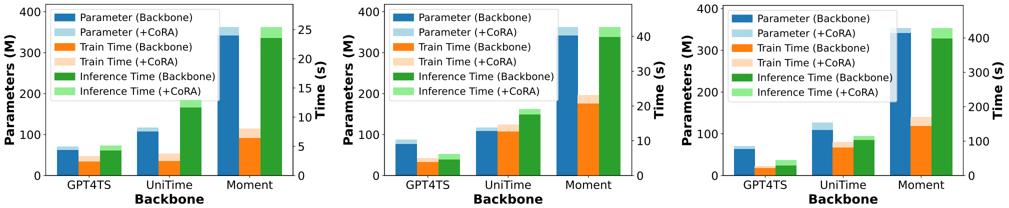
| Dataset | | | | ETTm2 | | | Electricity | | | Confidence Interval |
|---|---|---|---|---|---|---|---|---|---|---|
| | DCE | HD | HPCL | GPT4TS | UniTime | Timer | GPT4TS | UniTime | Timer | |
| 1 | ✗ | ✗ | ✗ | 0.279±0.002 | 0.293±0.003 | 0.262±0.003 | 0.207±0.003 | 0.202±0.003 | 0.242±0.002 | 99% |
| 2 | ○ | ○ | ✓ | 0.277±0.002 | 0.287±0.002 | 0.259±0.002 | 0.206±0.001 | 0.197±0.002 | 0.237±0.002 | 99% |
| 3 | ○ | ✓ | ✓ | 0.274±0.002 | 0.284±0.003 | 0.256±0.002 | 0.204±0.001 | 0.195±0.001 | 0.235±0.002 | 95% |
| 4 | ✓ | ○ | ✓ | 0.271±0.001 | 0.282±0.002 | 0.254±0.002 | 0.203±0.001 | 0.196±0.002 | 0.234±0.002 | 99% |
| 5 | ✓ | ✓ | ✓ | **0.267**±0.001 | **0.278**±0.002 | **0.250**±0.001 | **0.201**±0.001 | **0.191**±0.001 | **0.229**±0.002 | 99% |

✗ denotes a module removed, ✓ denotes a module added, ○ denotes replace a module with a naive implementation.

Specifically, we replace the DCE module with a series-level Pearson correlation coefficient, which cannot model DCorr, and the HD module with a single-branch projection layer, which is unable to capture PCorr. The comparison between Row 1-2 demonstrates the effectiveness of HPCL; however, its performance is limited due to its inability to capture multiple types of correlations. In Rows 2-4, the addition of either the DCE or HD module to HPCL further enhances performance, which confirms the efficacy of both modules. In Row 5, the combination of all three modules achieves the best performance.

### 6.5 Model Analysis

**Efficiency Analysis** Our proposed CoRA, as a lightweight plugin for TSFMs, shows strong efficiency, particularly during the inference phase. Figure 5 shows a comparative analysis of the efficiency of TSFMs with and without the application of CoRA. We selected three datasets in ascending order of the number of channels: ETTm2 ($N = 7$), Weather ($N = 21$), and Electricity ($N = 321$). For the experiments, both the look-back window $L$ and the forecasting horizon $H$ were set to 96. Train time and Inference time refer to the duration of a single training epoch and the total time required to process all samples during inference, respectively. The results show that, compared to the backbone itself, the use of CoRA does not introduce significant additional time or parameter numbers. Moreover, as the number of channels ($N$) increases, CoRA maintains its efficiency without noticeable degradation compared to the backbone, particularly during inference.



(a) ETTm2

(b) Weather

(c) Electricity

Figure 5: Efficiency Analysis of TSFMs with and without the application of CoRA.

**Data Analysis** While the previous results focused on the 5% fine-tuning setting, we expand the analysis to provide a more comprehensive view and to explicitly explore the impact of fine-tuning data volume on performance. Specifically, we fine-tune the TTM and CALF backbones on the ETTm2 and Weather datasets, using 3%, 5%, 10%, and 20% of the available training data. The MSE results are summarized in the Table 12. As the results indicate, CorA still yields a modest performance improvement even in a low-data regime using only 3% of the data.

Table 3: MSE results for different data percentage.

| **Dataset** | **ETTm2** | | | | **Weather** | | | |
|---|---|---|---|---|---|---|---|---|
| **Data** | **3%** | **5%** | **10%** | **20%** | **3%** | **5%** | **10%** | **20%** |
| **TTM** | 0.263<br>±.005 | 0.259<br>±.003 | 0.256<br>±.003 | 0.250<br>±.002 | 0.237<br>±.003 | 0.226<br>±.002 | 0.224<br>±.002 | 0.216<br>±.001 |
| **+ CorA** | **0.261**<br>±.004 | **0.249**<br>±.002 | **0.248**<br>±.001 | **0.245**<br>±.001 | **0.234**<br>±.003 | **0.214**<br>±.002 | **0.212**<br>±.001 | **0.210**<br>±.001 |
| **CALF** | 0.285<br>±.005 | 0.274<br>±.003 | 0.268<br>±.004 | 0.261<br>±.003 | 0.251<br>±.004 | 0.238<br>±.002 | 0.230<br>±.003 | 0.224<br>±.002 |
| **+ CorA** | **0.283**<br>±.003 | **0.263**<br>±.002 | **0.260**<br>±.002 | **0.254**<br>±.002 | **0.248**<br>±.003 | **0.229**<br>±.002 | **0.223**<br>±.002 | **0.219**<br>±.002 |

**Sensitivity Analysis and Visualization** The Data Sensitivity of CoRA in different few-shot setting are presented in Appendix F.1. The Prarmeter Sensitivity analyses for the polynomial's degree $K$, the decomposition size $M$, and the number of projection layers $N_1, N_2$ are presented in Appendix F.2. The Visualization of heterogeneous spaces are presented in Appendix F.3.

## 7 Conclusion

In this paper, we propose a lightweight Correlation-Aware Adapter (CoRA) that enhances the predictive performance of Time Series Foundation Models (TSFMs) by considering the mentioned three types of correlation relationships. Comprehensive experiments on real-world datasets demonstrate that CoRA can improve the forecast performance of TSFMs.

## Ethics statement

Our work exclusively uses publicly available benchmark datasets that contain no personally identifiable information. The proposed adapter for Time Series Foundation Models in Multivariate Time Series Forecasting is designed for beneficial applications in system reliability and safety monitoring. No human subjects were involved in this research.

## Reproducibility statement

The performance of CoRA and the datasets used in our work are real, and all experimental results can be reproduced. We have released our model code in an anonymous repository: https://anonymous.4open.science/r/CoRA-D968. Once the paper is accepted, we will release the scripts for all settings.

## References

Abdul Fatir Ansari, Lorenzo Stella, Caner Turkmen, Xiyuan Zhang, Pedro Mercado, Huibin Shen, Oleksandr Shchur, Syama Syndar Rangapuram, Sebastian Pineda Arango, Shubham Kapoor, Jasper Zschiegner, Danielle C. Maddix, Michael W. Mahoney, Kari Torkkola, Andrew Gordon Wilson, Michael Bohlke-Schneider, and Yuyang Wang. Chronos: Learning the language of time series. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856.

Wanlin Cai, Yuxuan Liang, Xianggen Liu, Jianshuai Feng, and Yuankai Wu. Msgnet: Learning multi-scale inter-series correlations for multivariate time series forecasting. In *AAAI*, pp. 11141–11149, 2024.

Jialin Chen, Jan Eric Lenssen, Aosong Feng, Weihua Hu, Matthias Fey, Leandros Tassiulas, Jure Leskovec, and Rex Ying. From similarity to superiority: Channel clustering for time series forecasting. In *NeurIPS*, 2024.

Razvan-Gabriel Cirstea, Tung Kieu, Chenjuan Guo, Bin Yang, and Sinno Jialin Pan. Enhancenet: Plugin neural networks for enhancing correlated time series forecasting. In *ICDE*, pp. 1739–1750, 2021.

Abhimanyu Das, Weihao Kong, Rajat Sen, and Yichen Zhou. A decoder-only foundation model for time-series forecasting. *arXiv preprint arXiv:2310.10688*, 2023.

Vijay Ekambaram, Arindam Jati, Nam Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. Tsmixer: Lightweight mlp-mixer model for multivariate time series forecasting. In *SIGKDD*, 2023.

Vijay Ekambaram, Arindam Jati, Nam H Nguyen, Pankaj Dayama, Chandra Reddy, Wesley M Gifford, and Jayant Kalagnanam. Ttms: Fast multi-level tiny time mixers for improved zero-shot and few-shot forecasting of multivariate time series. *arXiv preprint arXiv:2401.03955*, 2024a.

Vijay Ekambaram, Arindam Jati, Nam H. Nguyen, Pankaj Dayama, Chandra Reddy, Wesley M. Gifford, and Jayant Kalagnanam. Tiny time mixers (ttms): Fast pre-trained models for enhanced zero/few-shot forecasting of multivariate time series. *CoRR*, abs/2401.03955, 2024b.

Shanghua Gao, Teddy Koker, Owen Queen, Thomas Hartvigsen, Theodoros Tsiligkaridis, and Marinka Zitnik. Units: Building a unified time series model. *arXiv preprint arXiv:2403.00131*, 2024.

Mononito Goswami, Konrad Szafer, Arjun Choudhry, Yifu Cai, Shuo Li, and Artur Dubrawski. Moment: A family of open time-series foundation models. *arXiv preprint arXiv:2402.03885*, 2024.

Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7132–7141, 2018.

Qihe Huang, Lei Shen, Ruixin Zhang, Shouhong Ding, Binwu Wang, Zhengyang Zhou, and Yang Wang. Crossgnn: Confronting noisy multivariate time series via cross interaction refinement. In *NeurIPS*, 2023.

Maowei Jiang, Pengyu Zeng, Kai Wang, Huan Liu, Wenbo Chen, and Haoran Liu. Fecam: Frequency enhanced channel attention mechanism for time series forecasting. *Advanced Engineering Informatics*, 58:102158, 2023.

Ming Jin, Shiyu Wang, Lintao Ma, Zhixuan Chu, James Y Zhang, Xiaoming Shi, Pin-Yu Chen, Yuxuan Liang, Yuan-Fang Li, Shirui Pan, et al. Time-llm: Time series forecasting by reprogramming large language models. *arXiv preprint arXiv:2310.01728*, 2023.

Md Kowsher, Md Shohanur Islam Sobuj, Nusrat Jahan Prottasha, E Alejandro Alanis, Ozlem Ozmen Garibay, and Niloofar Yousefi. Llm-mixer: Multiscale mixing in llms for time series forecasting. *arXiv preprint arXiv:2410.11674*, 2024.

Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. Modeling long-and short-term temporal patterns with deep neural networks. In *SIGIR*, pp. 95–104, 2018.

Zhe Li, Xiangfei Qiu, Peng Chen, Yihang Wang, Hanyin Cheng, Yang Shu, Jilin Hu, Chenjuan Guo, Aoying Zhou, Qingsong Wen, et al. Fm4ts-bench: A comprehensive and unified benchmark of foundation models for time series forecasting. In *SIGKDD*, 2025.

Peiyuan Liu, Hang Guo, Tao Dai, Naiqi Li, Jigang Bao, Xudong Ren, Yong Jiang, and Shu-Tao Xia. Calf: Aligning llms for time series forecasting via cross-modal fine-tuning. *arXiv preprint arXiv:2403.07300*, 2024a.

Qinshuo Liu, Yanwen Fang, Pengtao Jiang, and Guodong Li. Dgcformer: Deep graph clustering transformer for multivariate time series forecasting. *arXiv preprint arXiv:2405.08440*, 2024b.

Xu Liu, Junfeng Hu, Yuan Li, Shizhe Diao, Yuxuan Liang, Bryan Hooi, and Roger Zimmermann. Unitime: A language-empowered unified model for cross-domain time series forecasting. In *Proceedings of the ACM on Web Conference*, 2024c.

Yijing Liu, Qinxian Liu, Jian-Wei Zhang, Haozhe Feng, Zhongwei Wang, Zihan Zhou, and Wei Chen. Multivariate time-series forecasting with temporal polynomial graph neural networks. *Advances in neural information processing systems*, 35:19414–19426, 2022.

Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. itransformer: Inverted transformers are effective for time series forecasting. *arXiv preprint arXiv:2310.06625*, 2023.

Yong Liu, Guo Qin, Xiangdong Huang, Jianmin Wang, and Mingsheng Long. Autotimes: Autoregressive time series forecasters via large language models. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024d.

Yong Liu, Haoran Zhang, Chenyu Li, Xiangdong Huang, Jianmin Wang, and Mingsheng Long. Timer: Transformers for time series analysis at scale. *arXiv preprint arXiv:2402.02368*, 2024e.

Tong Nie, Yuewen Mei, Guoyang Qin, Jian Sun, and Wei Ma. Channel-aware low-rank adaptation in time series forecasting. In *CIKM*, pp. 3959–3963, 2024.

Zijie Pan, Yushan Jiang, Sahil Garg, Anderson Schneider, Yuriy Nevmyvaka, and Dongjin Song. $s^2$ ip-llm: Semantic space informed prompt learning with llm for time series forecasting. In *Forty-first International Conference on Machine Learning*, 2024.

Rafael Poyatos, Víctor Granda, Víctor Flo, Mark A Adams, Balázs Adorján, David Aguadé, Marcos PM Aidar, Scott Allen, M Susana Alvarado-Barrientos, Kristina J Anderson-Teixeira, et al. Global transpiration data from sap flow measurements: the sapfluxnet database. *Earth System Science Data Discussions*, 2020:1–57, 2020.

Xiangfei Qiu, Jilin Hu, Lekui Zhou, Xingjian Wu, Junyang Du, Buang Zhang, Chenjuan Guo, Aoying Zhou, Christian S. Jensen, Zhenli Sheng, and Bin Yang. TFB: towards comprehensive and fair benchmarking of time series forecasting methods. *Proc. VLDB Endow.*, 17(9):2363–2377, 2024a.

Xiangfei Qiu, Jilin Hu, Lekui Zhou, Xingjian Wu, Junyang Du, Buang Zhang, Chenjuan Guo, Aoying Zhou, Christian S. Jensen, Zhenli Sheng, and Bin Yang. Tfb: Towards comprehensive and fair benchmarking of time series forecasting methods. *Proc. VLDB Endow.*, 17(9):2363–2377, 2024b.

Xiangfei Qiu, Hanyin Cheng, Xingjian Wu, Jilin Hu, Chenjuan Guo, and Bin Yang. A comprehensive survey of deep learning for multivariate time series forecasting: A channel strategy perspective, 2025a. URL `https://arxiv.org/abs/2502.10721`.

Xiangfei Qiu, Zhe Li, Wanghui Qiu, Shiyan Hu, Lekui Zhou, Xingjian Wu, Zhengyu Li, Chenjuan Guo, Aoying Zhou, Zhenli Sheng, Jilin Hu, Christian S. Jensen, and Bin Yang. TAB: unified benchmarking of time series anomaly detection methods. *Proc. VLDB Endow.*, 18(9):2775–2789, 2025b.

Xiangfei Qiu, Xingjian Wu, Yan Lin, Chenjuan Guo, Jilin Hu, and Bin Yang. Duet: Dual clustering enhanced multivariate time series forecasting. In *SIGKDD*, 2025c.

Chao Shang, Jie Chen, and Jinbo Bi. Discrete graph structure learning for forecasting multiple time series. *arXiv preprint arXiv:2101.06861*, 2021.

Zongjiang Shang, Ling Chen, Binqing Wu, and Dongliang Cui. Ada-mshyper: Adaptive multi-scale hypergraph transformer for time series forecasting. *CoRR*, abs/2410.23992, 2024.

Xiaoming Shi, Shiyu Wang, Yuqi Nie, Dianqi Li, Zhou Ye, Qingsong Wen, and Ming Jin. Time-moe: Billion-scale time series foundation models with mixture of experts. *CoRR*, abs/2409.16040, 2024.

Artur Trindade. ElectricityLoadDiagrams20112014. UCI Machine Learning Repository, 2015. DOI: https://doi.org/10.24432/C58C86.

Yihang Wang, Yuying Qiu, Peng Chen, Kai Zhao, Yang Shu, Zhongwen Rao, Lujia Pan, Bin Yang, and Chenjuan Guo. Rose: Register assisted general time series forecasting with decomposed frequency learning. *arXiv preprint arXiv:2405.17478*, 2024a.

Yuxuan Wang, Haixu Wu, Jiaxiang Dong, Yong Liu, Mingsheng Long, and Jianmin Wang. Deep time series models: A comprehensive survey and benchmark. *arXiv preprint arXiv:2407.13278*, 2024b.

Gerald Woo, Chenghao Liu, Akshat Kumar, Caiming Xiong, Silvio Savarese, and Doyen Sahoo. Unified training of universal time series forecasting transformers. *arXiv preprint arXiv:2402.02592*, 2024.

Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Advances in Neural Information Processing Systems*, 34:22419–22430, 2021.

Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, and Chengqi Zhang. Graph wavenet for deep spatial-temporal graph modeling. In Sarit Kraus (ed.), *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pp. 1907–1913. ijcai.org, 2019.

Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, Xiaojun Chang, and Chengqi Zhang. Connecting the dots: Multivariate time series forecasting with graph neural networks. In *SIGKDD*, pp. 753–763, 2020.

Jiawen Zhang, Xumeng Wen, Zhenwei Zhang, Shun Zheng, Jia Li, and Jiang Bian. ProbTS: Benchmarking point and distributional forecasting across diverse prediction horizons. In *NeurIPS Datasets and Benchmarks Track*, 2024.

Shuyi Zhang, Bin Guo, Anlan Dong, Jing He, Ziping Xu, and Song Xi Chen. Cautionary tales on air-quality improvement in beijing. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 473(2205):20170457, 2017.

Yunhao Zhang and Junchi Yan. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In *ICLR*, 2022.

Kai Zhao, Chenjuan Guo, Yunyao Cheng, Peng Han, Miao Zhang, and Bin Yang. Multiple time series forecasting with dynamic graph modeling. *Proc. VLDB Endow.*, 17(4):753–765, 2023.

Lifan Zhao and Yanyan Shen. Rethinking channel dependence for multivariate time series forecasting: Learning from leading indicators. In *ICLR*, 2024.

Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *AAAI*, volume 35, pp. 11106–11115, 2021.

Tian Zhou, Peisong Niu, Liang Sun, Rong Jin, et al. One fits all: Power general time series analysis by pretrained lm. *Advances in neural information processing systems*, 2023.

# A Definitions of the Three Correlations

**Definition 1** (*Correlation Martix*) Given a multivariate time series $X \in \mathbb{R}^{N \times L}$, where $N$ represents the number of channels and $L$ the temporal length, the series is partitioned into a set of $K = \lfloor L/T \rfloor$ non-overlapping segments using a window of length $T$. For the $k$-th segment, we define the channel-wise correlation matrix as $C^{(k)} \in \mathbb{R}^{N \times N}$, where $k \in [1, K]$. The element at the i-th row and j-th column of this matrix is denoted by $C_{ij}^{(k)}$ which represents the correlation between the i-th channel and the j-th channel.

**Definition 2** (*Dynamic Correlation*) The *Dynamic Correlation* refers to the case where the correlation martix $C^{(k)}(k \in [1, K])$ is not constant over the segments index $k$. Formally, this means there exist at least two distinct segment indices $m, n \in [1, K]$ with $m \neq n$, such that their corresponding correlation matrices are unequal: $C^{(m)} \neq C^{(n)}$.

**Definition 3** (*Heterogeneous Correlation*) The *Heterogeneous Correlation* refers to the case where there exists at least one temporal segment $k \in [1, K]$ in which some channels exhibit both positive and negative correlations with other channels. Formally, this means that for at least one correlation matrix $C^{(k)}(k \in [1, K])$, there exist at least three distinct channel indices $a, b, c \in [1, N]$, such that the corresponding correlation matrix elements $c_{ab}^{(k)}$ and $c_{ac}^{(k)}$ have opposite signs: $c_{ab}^{(k)} \cdot c_{ac}^{(k)} < 0$.

**Definition 4** (*Partial Correlation*) The *Partial Correlation* refers to the case where, within at least one temporal segment $k \in [1, K]$, there exists some pairs of channels with a non-significant relationship. Formally, given a predefined significance threshold $\epsilon$, this means there exists at least one matrix $C^{(k)}(k \in [1, K])$ and at least one pair of distinct channel indices $a, b \in [1, N]$, such that: $|c_{ab}^{(k)}| < \epsilon$.

# B Complexity Analyses

## B.1 Training Phase

**Dynamic Correlation Estimation** This module consits of Learnable Time-aware Polynomials (LTP) and Time-Varying and Time-Invariant (T-T) Composition. The LTP have a computational complexity of $\mathcal{O}(PKNM + PNd^2)$ due to the polynomial operations and the MLP used to generate $\boldsymbol{\mathcal{C}}_t$, and a space complexity of $\mathcal{O}(Kd + MN)$ because of the basis $\boldsymbol{q}$ and the MLP used to generate $\boldsymbol{\mathcal{C}}_t$. Where $P$ is the number of patches, $N$ denotes the number of channels, $M$ is the second dimension of $\boldsymbol{Q}_t$ ,$K$ is the degree of the polynomial and $d$ is the dimension of representations. The T-T Composition has a computational complexity of $\mathcal{O}(lN^2 + NM^2 + MN^2)$ due to the calculation of the Pearson coefficient and the composition in Equation 5. Where $l$ denotes the patch size. **Heterogeneous Division** This module has a computational complexity of $\mathcal{O}(NP^2d^2)$ and a space complexity of $\mathcal{O}(Pd)$ due to the channel-aware projection. **H-PCorr Contrastive Learning** The time complexity of calculating loss in Equation 11 is $\mathcal{O}(PdN^2)$.

Since $P$, $K$, $M$, and $l$ are much smaller than $N$ and $d$, they will not be considered as the primary components in the complexity analysis. So the total computational complexity is $\mathcal{O}(dN^2 + Nd^2)$ and the total space complexity is $\mathcal{O}(d + N)$. Most models cannot avoid having a computational complexity of $\mathcal{O}(d^2)$ and a space complexity of $\mathcal{O}(d)$. Therefore, if we focus only on the complexity with respect to (N) in our discussion, our model has a computational complexity of $\mathcal{O}(N^2)$ and a space complexity of $\mathcal{O}(N)$ during training.

## B.2 Inference Phase

In inference Phase, CoRA only includes projectors in the Heterogeneous Division and Heterogeneous Fusion modules, based on the above discussion, our model has a computational complexity of $\mathcal{O}(N)$ and a space complexity of $\mathcal{O}(1)$ during inference.

# C Theoretical Analyses

## C.1 The significance of Time-Varying and Time-Invariant Composition

The channel correlation can be expressed by combining a long-term stable state with dynamic changes. We decompose the learnable correlation into two parts, $\boldsymbol{Q}_t$ and $\boldsymbol{V}$, as shown in Figure 3, to fit the correlation relationship lightweightly.

**Theorem 1** *When the time series is locally stationary, the Time-Varying and Time-Invariant Decomposition allows $\boldsymbol{Q}_t\boldsymbol{V}\boldsymbol{Q}_t^T$ to contain both time-varying and time-invariant information, like conventional additive decomposition.*

*Specifically, $\boldsymbol{Q}_t\boldsymbol{V}\boldsymbol{Q}_t^T$ can be expressed as the sum of a time-invariant matrix $\boldsymbol{M}_i$ and a time-varying matrix $\boldsymbol{M}_v$, as shown below:*

$$\boldsymbol{Q}_t\boldsymbol{V}\boldsymbol{Q}_t^T = \boldsymbol{M}_i + \boldsymbol{M}_v \ . \tag{17}$$

*Proof.* Under the assumptions of locally stationary, $\boldsymbol{Q}_t$ can be expressed as $\bar{\boldsymbol{Q}}_t + \tilde{\boldsymbol{Q}}_t$, where $\bar{\boldsymbol{Q}}_t$ represents the mean value and $\tilde{\boldsymbol{Q}}_t$represents the residual. Therefore,$\boldsymbol{M}_t^{corr}$can be expressed as:

$$\begin{aligned}
\boldsymbol{Q}_t\boldsymbol{V}\boldsymbol{Q}_t^T &= (\bar{\boldsymbol{Q}}_t + \tilde{\boldsymbol{Q}}_t)\boldsymbol{V}(\bar{\boldsymbol{Q}}_t + \tilde{\boldsymbol{Q}}_t)^T \\
&= \bar{\boldsymbol{Q}}_t\boldsymbol{V}\bar{\boldsymbol{Q}}_t^T + \bar{\boldsymbol{Q}}_t\boldsymbol{V}\tilde{\boldsymbol{Q}}_t^T + \tilde{\boldsymbol{Q}}_t\boldsymbol{V}\bar{\boldsymbol{Q}}_t^T + \tilde{\boldsymbol{Q}}_t\boldsymbol{V}\tilde{\boldsymbol{Q}}_t^T \\
&= (\bar{\boldsymbol{Q}}_t\boldsymbol{V}\bar{\boldsymbol{Q}}_t^T) + (\bar{\boldsymbol{Q}}_t\boldsymbol{V}\tilde{\boldsymbol{Q}}_t^T + \tilde{\boldsymbol{Q}}_t\boldsymbol{V}\bar{\boldsymbol{Q}}_t^T + \tilde{\boldsymbol{Q}}_t\boldsymbol{V}\tilde{\boldsymbol{Q}}_t^T) \\
&= \boldsymbol{M}_i + \boldsymbol{M}_v \ ,
\end{aligned} \tag{18}$$

where $\boldsymbol{M}_i = \bar{\boldsymbol{Q}}_t\boldsymbol{V}\bar{\boldsymbol{Q}}_t^T$ has the same value at different times, and $\boldsymbol{M}_v = \bar{\boldsymbol{Q}}_t\boldsymbol{V}\tilde{\boldsymbol{Q}}_t^T + \tilde{\boldsymbol{Q}}_t\boldsymbol{V}\bar{\boldsymbol{Q}}_t^T + \tilde{\boldsymbol{Q}}_t\boldsymbol{V}\tilde{\boldsymbol{Q}}_t^T$ has different values at different times.

## C.2 The fitting ability of Time-aware Polynomials

Since time series exhibit regular changes, such as trends and seasonality, the dynamic correlation changes also have a certain regularity. To this end, we propose Time-aware Polynomials to fit the changing correlations better.

**Theorem 2** *When the time series is locally stationary, we can approximate the underlying correlation matrix with a high-order polynomial.*

*Specifically, assuming that the correlation is a smooth function to the basis $\boldsymbol{q}$, the true correlation component $\boldsymbol{Q}_t^*$ can be expressed as $\mathcal{F}(\boldsymbol{q})$. The fitting error of Time-aware Polynomials decreases as the highest degree $K$ of the polynomial increases. The error can be formalized as follows:*

$$|\boldsymbol{Q}_t^* - \boldsymbol{Q}_t| = \frac{\mathcal{F}^{(K+1)}(\boldsymbol{\xi})}{(K+1)!}\boldsymbol{q}^{(K+1)}, \boldsymbol{\xi} \in [-|\boldsymbol{q}|, |\boldsymbol{q}|] \ . \tag{19}$$

*Proof.* Given the true correlation as $\mathcal{F}(\boldsymbol{q})$, Since $\mathcal{F}$ is sufficiently smooth to the basis $\boldsymbol{q}$, we can perform a Maclaurin expansion of $\mathcal{F}$ around $\boldsymbol{0}$:

$$\mathcal{F}(\boldsymbol{q}) = \mathcal{F}(\boldsymbol{0}) + \mathcal{F}'(\boldsymbol{0})\boldsymbol{q} + \frac{\mathcal{F}''(\boldsymbol{0})}{2!}\boldsymbol{q}^2 + \cdots + \frac{\mathcal{F}^{(K)}(\boldsymbol{0})}{K!}\boldsymbol{q}^K + \cdots + \frac{\mathcal{F}^{(n)}(\boldsymbol{0})}{n!}\boldsymbol{q}^n + \cdots \ . \tag{20}$$

We construct auxiliary functions:

$$\mathcal{H}(\boldsymbol{t}) = \mathcal{F}(\boldsymbol{q}) - [\mathcal{F}(\boldsymbol{t}) + \mathcal{F}'(\boldsymbol{t})(\boldsymbol{q}-\boldsymbol{t}) + \frac{\mathcal{F}''(\boldsymbol{t})}{2!}(\boldsymbol{q}-\boldsymbol{t})^2 + \cdots + \frac{\mathcal{F}^{(K)}(\boldsymbol{t})}{K!}(\boldsymbol{q}-\boldsymbol{t})^K] \ , \tag{21}$$

$$\mathcal{G}(\boldsymbol{t}) = (\boldsymbol{q}-\boldsymbol{t})^{(K+1)} \ . \tag{22}$$

Assume $\boldsymbol{q} > 0$. Then, $\mathcal{H}$ and $\mathcal{G}$ are still continuously differentiable, and the following rules apply:

$$\mathcal{H}(\boldsymbol{t})' = -\frac{\mathcal{F}^{(K+1)}(\boldsymbol{t})}{K!}(\boldsymbol{q}-\boldsymbol{t})^K \ , \tag{23}$$

$$\mathcal{G}(\boldsymbol{t})' = -(K+1)(\boldsymbol{q}-\boldsymbol{t})^K \neq \boldsymbol{0} \ . \tag{24}$$

Since $\mathcal{H}(\boldsymbol{q}) = \mathcal{G}(\boldsymbol{q}) = 0$, by the Cauchy Mean Value Theorem, $\exists\, \boldsymbol{\xi} \in (0, \boldsymbol{q}), s.t.$

$$\frac{\mathcal{H}(\mathbf{0})}{\mathcal{G}(\mathbf{0})} = \frac{\mathcal{H}(\mathbf{0}) - \mathcal{H}(\boldsymbol{q})}{\mathcal{G}(\mathbf{0}) - \mathcal{G}(\boldsymbol{q})} = \frac{\mathcal{H}(\mathbf{0}) - \mathcal{H}(\boldsymbol{q})}{\mathcal{G}(\mathbf{0}) - \mathcal{G}(\boldsymbol{q})} = \frac{\mathcal{H}'(\boldsymbol{\xi})}{\mathcal{G}'(\boldsymbol{\xi})} = \frac{\mathcal{F}^{(K+1)}(\boldsymbol{\xi})}{(K+1)!}\ . \tag{25}$$

Therefore, we can derive the following equation:

$$\mathcal{F}(\boldsymbol{q}) = \mathcal{F}(\boldsymbol{0}) + \mathcal{F}'(\boldsymbol{0})\boldsymbol{q} + \frac{\mathcal{F}''(\boldsymbol{0})}{2!}\boldsymbol{q}^2 + \cdots + \frac{\mathcal{F}^{(K)}(\boldsymbol{0})}{K!}\boldsymbol{q}^K + \frac{\mathcal{F}^{(K+1)}(\boldsymbol{\xi})}{(K+1)!}\boldsymbol{q}^{(K+1)}, \boldsymbol{\xi} \in [0, \boldsymbol{q}]\ . \tag{26}$$

Let $C_{i,t} = \frac{\mathcal{F}^{(i)}}{i!}$. Then, we have the following equation:

$$\mathcal{F}(\boldsymbol{q}) = C_{0,t} + C_{1,t}\boldsymbol{q} + C_{2,t}\boldsymbol{q}^2 + \cdots + C_{K,t}\boldsymbol{q}^K + \frac{\mathcal{F}^{(K+1)}(\boldsymbol{\xi})}{(K+1)!}\boldsymbol{q}^{(K+1)}, \boldsymbol{\xi} \in [0, \boldsymbol{q}]\ . \tag{27}$$

That is:

$$\mathcal{F}(\boldsymbol{q}) - \boldsymbol{Q_t} = \frac{\mathcal{F}^{(K+1)}(\boldsymbol{\xi})}{(K+1)!}\boldsymbol{q}^{(K+1)}, \boldsymbol{\xi} \in [0, \boldsymbol{q}]\ . \tag{28}$$

For all $\boldsymbol{q} > 0$ and $\boldsymbol{q} < 0$, we have the following equation:

$$|\mathcal{F}(\boldsymbol{q}) - \boldsymbol{Q_t}| = \frac{\mathcal{F}^{(K+1)}(\boldsymbol{\xi})}{(K+1)!}\boldsymbol{q}^{(K+1)}, \boldsymbol{\xi} \in [-|\boldsymbol{q}|, |\boldsymbol{q}|]\ . \tag{29}$$

And that is:

$$|\boldsymbol{Q_t^*} - \boldsymbol{Q_t}| = \frac{\mathcal{F}^{(K+1)}(\boldsymbol{\xi})}{(K+1)!}\boldsymbol{q}^{(K+1)}, \boldsymbol{\xi} \in [-|\boldsymbol{q}|, |\boldsymbol{q}|]\ . \tag{30}$$

# D Experimental Details

## D.1 Datasets

To conduct comprehensive and fair comparisons for different models, we conduct experiments on ten well-known forecasting benchmarks as the target datasets, including: (I) **ETT** (Zhou et al., 2021) datasets contain 7 variates collected from two different electric transformers from July 2016 to July 2018. It consists of four subsets, of which ETTh1/ETTh2 are recorded hourly, and ETTm1/ETTm2 are recorded every 15 minutes. (II) **Electricity** (Trindade, 2015) contains the electricity consumption of 321 customers from July 2016 to July 2019, recorded hourly. (III) **Traffic** (Wu et al., 2021) contains road occupancy rates measured by 862 sensors on freeways in the San Francisco Bay Area from 2015 to 2016, recorded hourly. (IV) **Solar** (Lai et al., 2018) records solar power generation from 137 PV plants in 2006, every 10 minutes. (V) **Weather** (Wu et al., 2021) collects 21 meteorological indicators, including temperature and barometric pressure, for Germany in 2020, recorded every 10 minutes. (VI) **AQShunyi** (Zhang et al., 2017) is an air quality dataset from a measurement station, for 4 years. (VII) **ZafNoo** (Poyatos et al., 2020) is collected from the Sapflux data project and includes sap flow measurements and environmental variables. The details of the benchmark datasets are included in Table 4

Table 4: Statistics of datasets.

| Dataset | Domain | Frequency | Lengths | Dim | Split | Description |
|---|---|---|---|---|---|---|
| ETTh1 | Electricity | 1 hour | 14,400 | 7 | 6:2:2 | Power transformer 1, comprising seven indicators such as oil temperature and useful load |
| ETTh2 | Electricity | 1 hour | 14,400 | 7 | 6:2:2 | Power transformer 2, comprising seven indicators such as oil temperature and useful load |
| ETTm1 | Electricity | 15 mins | 57,600 | 7 | 6:2:2 | Power transformer 1, comprising seven indicators such as oil temperature and useful load |
| ETTm2 | Electricity | 15 mins | 57,600 | 7 | 6:2:2 | Power transformer 2, comprising seven indicators such as oil temperature and useful load |
| Weather | Environment | 10 mins | 52,696 | 21 | 7:1:2 | Recorded every for the whole year 2020, which contains 21 meteorological indicators |
| Electricity | Electricity | 1 hour | 26,304 | 321 | 7:1:2 | Electricity records the electricity consumption in kWh every 1 hour from 2012 to 2014 |
| Solar | Energy | 10 mins | 52,560 | 137 | 6:2:2 | Solar production records collected from 137 PV plants in Alabama |
| Traffic | Traffic | 1 hour | 17,544 | 862 | 7:1:2 | Road occupancy rates measured by 862 sensors on San Francisco Bay area freeways |
| AQShunyi | Environment | 1 hour | 35,064 | 11 | 7:1:2 | Air quality dataset from a measurement station, for 4 years |
| ZafNoo | Nature | 30 mins | 19,225 | 11 | 7:1:2 | Sap flow measurements and environmental variables from the Sapflux data project. |

## D.2 Baselines

In the realm of time series forecasting, numerous models have surfaced in recent years. We choose models with superior predictive performance in our benchmark, including the pre-trained time series models: Timer (Liu et al., 2024e), TTM (Ekambaram et al., 2024a) and Moment (Goswami et al., 2024); The LLM-based models: CALF (Liu et al., 2024a), GPT4TS (Zhou et al., 2023), UniTime (Liu et al., 2024c); The specific descriptions for each of these models—see Table 5.

Table 5: Descriptions of time series forecasting models in experiment.

| **Models** | **Descriptions** |
|---|---|
| Moment (Goswami et al., 2024) | Moment is a transformer system pre-trained on a masked time series task. It reconstructs masked portions of time series for tasks like forecasting, classification, anomaly detection, and imputation. |
| TTM (Ekambaram et al., 2024a) | It is based on MLP-Mixer blocks with gated attention and multi-resolution sampling. It captures temporal patterns and cross-channel correlations for time-series forecasting, optimized for zero/few-shot learning with low computational cost. |
| Timer (Liu et al., 2024e) | Timer is a GPT-style autoregressive model for time series analysis, predicting the next token in single-series sequences. It supports tasks like forecasting, imputation, and anomaly detection across different time series. |
| CALF (Liu et al., 2024a) | CALF is a cross-modal knowledge distillation framework that aligns time series data with pre-trained LLMs by leveraging both static and dynamic knowledge, achieving state-of-the-art performance in both long- and short-term forecasting tasks with strong generalization abilities. |
| GPT4TS (Zhou et al., 2023) | GPT4TS fine-tunes the limited parameters of LLM, which demonstrates competitive performance by transferring knowledge from large-scale pre-training text data. |
| UniTime (Liu et al., 2024c) | UniTime designs domain instructions to align time series and text modalities. |

## D.3 Implementation Details

We utilize the FM4TS-Bench (Li et al., 2025) code repository for unified evaluation. Following the settings in TFB (Qiu et al., 2024b) and FM4TS-Bench, we do not apply the Drop Last trick to ensure a fair comparison. All experiments of CoRA are conducted using PyTorch in Python 3.10 and executed on an NVIDIA Tesla-A800 GPU. The MSE loss function guides the training process and employs the ADAM optimizer.

# E Full results

Table 6: The table reports MSE and MAE of LLM-based models for different forecasting horizons $F \in \{96, 192, 336, 720\}$. The better results are highlighted in **bold**.

| Model | | GPT4TS (2023) | | | | CALF (2025) | | | | UniTime (2024) | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Plugin | | ✗ | | ✓ | | ✗ | | ✓ | | ✗ | | ✓ | |
| Metric | | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE |
| ETTh1 | 96 | 0.438 | 0.445 | **0.427** | **0.435** | 0.405 | 0.426 | **0.394** | **0.417** | 0.717 | 0.575 | **0.690** | **0.556** |
| | 192 | 0.460 | 0.458 | **0.449** | **0.448** | 0.428 | 0.442 | **0.418** | **0.433** | 0.750 | 0.591 | **0.723** | **0.572** |
| | 336 | 0.462 | 0.467 | **0.451** | **0.458** | 0.443 | 0.454 | **0.433** | **0.445** | 0.723 | 0.592 | **0.700** | **0.580** |
| | 720 | 0.509 | 0.511 | **0.497** | **0.496** | 0.495 | 0.494 | **0.486** | **0.483** | 0.765 | 0.622 | **0.739** | **0.606** |
| ETTh2 | 96 | 0.329 | 0.380 | **0.318** | **0.368** | 0.302 | 0.362 | **0.295** | **0.354** | 0.359 | 0.391 | **0.349** | **0.382** |
| | 192 | 0.368 | 0.406 | **0.356** | **0.393** | 0.385 | 0.400 | **0.377** | **0.391** | 0.388 | 0.428 | **0.376** | **0.416** |
| | 336 | 0.378 | 0.421 | **0.368** | **0.412** | 0.387 | 0.418 | **0.380** | **0.408** | 0.392 | 0.436 | **0.380** | **0.427** |
| | 720 | 0.418 | 0.450 | **0.404** | **0.440** | 0.416 | 0.449 | **0.409** | **0.456** | 0.454 | 0.472 | **0.435** | **0.457** |
| ETTm1 | 96 | 0.343 | 0.379 | **0.333** | **0.371** | 0.317 | 0.366 | **0.308** | **0.359** | 0.357 | 0.384 | **0.343** | **0.376** |
| | 192 | 0.375 | 0.398 | **0.366** | **0.389** | 0.346 | 0.380 | **0.337** | **0.371** | 0.386 | 0.401 | **0.370** | **0.390** |
| | 336 | 0.394 | 0.406 | **0.385** | **0.398** | 0.385 | 0.405 | **0.377** | **0.396** | 0.420 | 0.420 | **0.402** | **0.406** |
| | 720 | 0.440 | 0.434 | **0.432** | **0.425** | 0.439 | 0.433 | **0.432** | **0.423** | 0.468 | 0.446 | **0.458** | **0.435** |
| ETTm2 | 96 | 0.190 | 0.279 | **0.183** | **0.273** | 0.180 | 0.272 | **0.176** | **0.265** | 0.190 | 0.277 | **0.183** | **0.267** |
| | 192 | 0.241 | 0.312 | **0.232** | **0.305** | 0.236 | 0.310 | **0.228** | **0.302** | 0.248 | 0.315 | **0.238** | **0.305** |
| | 336 | 0.296 | 0.349 | **0.288** | **0.340** | 0.295 | 0.348 | **0.286** | **0.339** | 0.345 | 0.374 | **0.333** | **0.365** |
| | 720 | 0.385 | 0.401 | **0.371** | **0.389** | 0.372 | 0.397 | **0.363** | **0.390** | 0.380 | 0.392 | **0.367** | **0.381** |
| Electricity | 96 | 0.178 | 0.294 | **0.176** | **0.288** | 0.141 | 0.240 | **0.138** | **0.234** | 0.174 | 0.282 | **0.170** | **0.275** |
| | 192 | 0.192 | 0.306 | **0.186** | **0.302** | 0.156 | 0.254 | **0.151** | **0.247** | 0.185 | 0.291 | **0.180** | **0.284** |
| | 336 | 0.208 | 0.318 | **0.204** | **0.313** | 0.174 | 0.271 | **0.168** | **0.262** | 0.201 | 0.305 | **0.194** | **0.298** |
| | 720 | 0.248 | 0.348 | **0.241** | **0.339** | 0.216 | 0.306 | **0.212** | **0.299** | 0.240 | 0.335 | **0.232** | **0.325** |
| Traffic | 96 | 0.439 | 0.322 | **0.429** | **0.314** | 0.406 | 0.298 | **0.394** | **0.290** | 0.423 | 0.309 | **0.414** | **0.301** |
| | 192 | 0.422 | 0.304 | **0.413** | **0.298** | 0.423 | 0.309 | **0.412** | **0.300** | 0.435 | 0.319 | **0.425** | **0.309** |
| | 336 | 0.432 | 0.308 | **0.424** | **0.301** | 0.436 | 0.317 | **0.424** | **0.307** | 0.474 | 0.331 | **0.464** | **0.325** |
| | 720 | 0.468 | 0.325 | **0.454** | **0.315** | 0.477 | 0.340 | **0.467** | **0.332** | 0.485 | 0.362 | **0.476** | **0.353** |
| Solar | 96 | 0.242 | 0.261 | **0.233** | **0.251** | 0.203 | 0.274 | **0.198** | **0.269** | 0.249 | 0.284 | **0.244** | **0.278** |
| | 192 | 0.258 | 0.294 | **0.249** | **0.282** | 0.224 | 0.290 | **0.219** | **0.284** | 0.250 | 0.320 | **0.245** | **0.313** |
| | 336 | 0.258 | 0.278 | **0.249** | **0.267** | 0.243 | 0.308 | **0.238** | **0.301** | 0.253 | 0.322 | **0.246** | **0.315** |
| | 720 | 0.259 | 0.279 | **0.247** | **0.274** | 0.247 | 0.314 | **0.239** | **0.307** | 0.255 | 0.325 | **0.251** | **0.318** |
| Weather | 96 | 0.187 | 0.244 | **0.180** | **0.236** | 0.163 | 0.217 | **0.159** | **0.211** | 0.184 | 0.239 | **0.174** | **0.229** |
| | 192 | 0.225 | 0.274 | **0.217** | **0.264** | 0.206 | 0.253 | **0.200** | **0.247** | 0.227 | 0.274 | **0.216** | **0.265** |
| | 336 | 0.268 | 0.304 | **0.259** | **0.294** | 0.258 | 0.292 | **0.252** | **0.283** | 0.271 | 0.305 | **0.258** | **0.292** |
| | 720 | 0.330 | 0.348 | **0.320** | **0.335** | 0.322 | 0.339 | **0.312** | **0.330** | 0.334 | 0.350 | **0.321** | **0.334** |
| AQShunyi | 96 | 0.799 | 0.535 | **0.785** | **0.526** | 0.689 | 0.508 | **0.674** | **0.500** | 0.689 | 0.513 | **0.666** | **0.495** |
| | 192 | 0.846 | 0.549 | **0.823** | **0.538** | 0.720 | 0.515 | **0.699** | **0.502** | 0.737 | 0.521 | **0.711** | **0.509** |
| | 336 | 0.854 | 0.555 | **0.831** | **0.545** | 0.734 | 0.525 | **0.718** | **0.516** | 0.747 | 0.541 | **0.720** | **0.527** |
| | 720 | 0.897 | 0.573 | **0.883** | **0.558** | 0.784 | 0.551 | **0.769** | **0.537** | 0.796 | 0.564 | **0.766** | **0.544** |
| ZafNoo | 96 | 0.515 | 0.486 | **0.505** | **0.478** | 0.469 | 0.434 | **0.456** | **0.424** | 0.472 | 0.447 | **0.457** | **0.434** |
| | 192 | 0.552 | 0.505 | **0.541** | **0.494** | 0.532 | 0.475 | **0.516** | **0.462** | 0.547 | 0.482 | **0.525** | **0.464** |
| | 336 | 0.582 | 0.515 | **0.574** | **0.507** | 0.567 | 0.493 | **0.553** | **0.482** | 0.571 | 0.496 | **0.551** | **0.483** |
| | 720 | 0.610 | 0.532 | **0.593** | **0.516** | 0.628 | 0.521 | **0.610** | **0.507** | 0.658 | 0.538 | **0.633** | **0.521** |

Table 7: The table reports MSE and MAE of pre-trained models for different forecasting horizons $F \in \{96, 192, 336, 720\}$. The better results are highlighted in **bold**.

| Model | | Timer (2024) | | | | Moment (2024) | | | | TTM (2024) | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Plugin | | ✗ | | ✓ | | ✗ | | ✓ | | ✗ | | ✓ | |
| Metric | | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE |
| ETTh1 | 96 | 0.394 | 0.408 | **0.384** | **0.401** | 0.408 | 0.422 | **0.398** | **0.413** | 0.363 | 0.392 | **0.355** | **0.380** |
| | 192 | 0.456 | 0.458 | **0.446** | **0.450** | 0.428 | 0.436 | **0.418** | **0.425** | 0.391 | 0.409 | **0.379** | **0.401** |
| | 336 | 0.495 | 0.487 | **0.483** | **0.480** | 0.456 | 0.450 | **0.445** | **0.439** | 0.411 | 0.429 | **0.398** | **0.420** |
| | 720 | 0.849 | 0.651 | **0.835** | **0.633** | 0.482 | 0.482 | **0.468** | **0.465** | 0.453 | 0.471 | **0.439** | **0.458** |
| ETTh2 | 96 | 0.291 | 0.342 | **0.282** | **0.335** | 0.310 | 0.360 | **0.299** | **0.346** | 0.271 | 0.329 | **0.264** | **0.323** |
| | 192 | 0.371 | 0.395 | **0.358** | **0.388** | 0.347 | 0.387 | **0.337** | **0.374** | 0.339 | 0.373 | **0.329** | **0.365** |
| | 336 | 0.371 | 0.413 | **0.360** | **0.403** | 0.365 | 0.406 | **0.351** | **0.391** | 0.372 | 0.401 | **0.362** | **0.394** |
| | 720 | 0.441 | 0.465 | **0.429** | **0.454** | 0.401 | 0.436 | **0.384** | **0.423** | 0.385 | 0.428 | **0.376** | **0.415** |
| ETTm1 | 96 | 0.302 | 0.349 | **0.292** | **0.341** | 0.311 | 0.356 | **0.303** | **0.346** | 0.299 | 0.343 | **0.291** | **0.333** |
| | 192 | 0.363 | 0.389 | **0.351** | **0.382** | 0.341 | 0.374 | **0.329** | **0.363** | 0.341 | 0.367 | **0.329** | **0.355** |
| | 336 | 0.405 | 0.412 | **0.394** | **0.405** | 0.367 | 0.389 | **0.355** | **0.379** | 0.365 | 0.381 | **0.355** | **0.369** |
| | 720 | 0.749 | 0.560 | **0.723** | **0.542** | 0.415 | 0.416 | **0.401** | **0.412** | 0.420 | 0.412 | **0.405** | **0.405** |
| ETTm2 | 96 | 0.168 | 0.248 | **0.163** | **0.243** | 0.175 | 0.263 | **0.169** | **0.255** | 0.164 | 0.250 | **0.160** | **0.244** |
| | 192 | 0.237 | 0.301 | **0.230** | **0.293** | 0.226 | 0.297 | **0.218** | **0.288** | 0.222 | 0.290 | **0.214** | **0.283** |
| | 336 | 0.321 | 0.362 | **0.310** | **0.353** | 0.278 | 0.332 | **0.267** | **0.324** | 0.282 | 0.330 | **0.271** | **0.320** |
| | 720 | 0.385 | 0.413 | **0.375** | **0.404** | 0.362 | 0.387 | **0.350** | **0.378** | 0.364 | 0.381 | **0.354** | **0.369** |
| Electricity | 96 | 0.139 | 0.235 | **0.136** | **0.230** | 0.158 | 0.242 | **0.151** | **0.233** | 0.146 | 0.246 | **0.143** | **0.239** |
| | 192 | 0.162 | 0.255 | **0.159** | **0.249** | 0.186 | 0.264 | **0.178** | **0.254** | 0.165 | 0.264 | **0.159** | **0.256** |
| | 336 | 0.183 | 0.280 | **0.180** | **0.274** | 0.256 | 0.286 | **0.247** | **0.277** | 0.181 | 0.281 | **0.174** | **0.272** |
| | 720 | 0.319 | 0.366 | **0.312** | **0.358** | 0.359 | 0.372 | **0.348** | **0.358** | 0.223 | 0.315 | **0.216** | **0.305** |
| Traffic | 96 | 0.381 | 0.272 | **0.372** | **0.266** | 0.391 | 0.278 | **0.380** | **0.268** | 0.448 | 0.324 | **0.433** | **0.313** |
| | 192 | 0.413 | 0.286 | **0.402** | **0.279** | 0.443 | 0.296 | **0.431** | **0.285** | 0.466 | 0.330 | **0.450** | **0.319** |
| | 336 | 0.434 | 0.298 | **0.422** | **0.292** | 0.436 | 0.302 | **0.422** | **0.290** | 0.491 | 0.345 | **0.473** | **0.336** |
| | 720 | 0.570 | 0.484 | **0.554** | **0.473** | 0.550 | 0.493 | **0.529** | **0.478** | 0.533 | 0.365 | **0.518** | **0.351** |
| Solar | 96 | 0.170 | 0.218 | **0.165** | **0.212** | 0.203 | 0.269 | **0.196** | **0.260** | 0.254 | 0.207 | **0.246** | **0.200** |
| | 192 | 0.197 | 0.247 | **0.191** | **0.239** | 0.215 | 0.275 | **0.208** | **0.266** | 0.270 | 0.240 | **0.262** | **0.232** |
| | 336 | 0.203 | 0.253 | **0.195** | **0.246** | 0.223 | 0.281 | **0.215** | **0.273** | 0.274 | 0.239 | **0.266** | **0.231** |
| | 720 | 0.336 | 0.335 | **0.325** | **0.326** | 0.225 | 0.281 | **0.216** | **0.272** | 0.277 | 0.237 | **0.266** | **0.231** |
| Weather | 96 | 0.150 | 0.199 | **0.146** | **0.194** | 0.168 | 0.225 | **0.144** | **0.193** | 0.147 | 0.195 | **0.145** | **0.189** |
| | 192 | 0.214 | 0.264 | **0.207** | **0.257** | 0.210 | 0.259 | **0.208** | **0.256** | 0.194 | 0.238 | **0.187** | **0.230** |
| | 336 | 0.282 | 0.316 | **0.274** | **0.309** | 0.255 | 0.292 | **0.274** | **0.308** | 0.244 | 0.277 | **0.236** | **0.268** |
| | 720 | 0.360 | 0.374 | **0.349** | **0.362** | 0.326 | 0.342 | **0.350** | **0.365** | 0.314 | 0.329 | **0.295** | **0.316** |
| AQShunyi | 96 | 0.534 | 0.434 | **0.520** | **0.421** | 0.728 | 0.490 | **0.701** | **0.477** | 0.638 | 0.479 | **0.615** | **0.464** |
| | 192 | 0.712 | 0.520 | **0.690** | **0.503** | 0.706 | 0.509 | **0.683** | **0.495** | 0.687 | 0.501 | **0.666** | **0.488** |
| | 336 | 0.734 | 0.525 | **0.713** | **0.511** | 0.723 | 0.519 | **0.698** | **0.504** | 0.708 | 0.515 | **0.681** | **0.502** |
| | 720 | 0.791 | 0.541 | **0.763** | **0.527** | 0.776 | 0.543 | **0.755** | **0.532** | 0.765 | 0.543 | **0.742** | **0.524** |
| ZafNoo | 96 | 0.436 | 0.399 | **0.426** | **0.391** | 0.475 | 0.441 | **0.461** | **0.427** | 0.424 | 0.403 | **0.409** | **0.387** |
| | 192 | 0.522 | 0.452 | **0.511** | **0.439** | 0.521 | 0.464 | **0.505** | **0.445** | 0.484 | 0.441 | **0.467** | **0.426** |
| | 336 | 0.547 | 0.479 | **0.532** | **0.467** | 0.558 | 0.482 | **0.539** | **0.465** | 0.535 | 0.467 | **0.512** | **0.452** |
| | 720 | 0.615 | 0.513 | **0.603** | **0.500** | 0.593 | 0.500 | **0.568** | **0.484** | 0.569 | 0.492 | **0.547** | **0.475** |

# F More Analysis on CoRA

## F.1 Comparison in Different Sampling Rate

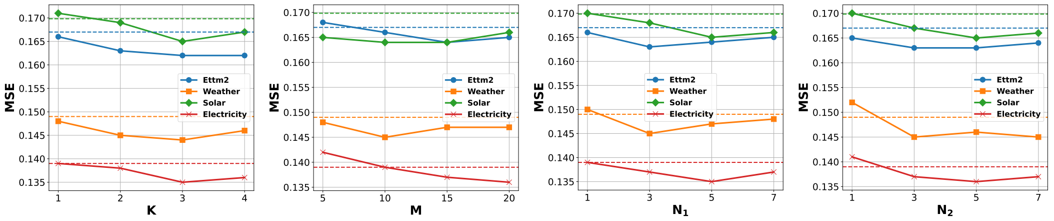
To further demonstrate the advantages of CoRA, we compared its performance with LIFT (Zhao & Shen, 2024) and C-LoRA (Nie et al., 2024) using TTM as the backbone and setting H to 96, under different sampling rates in the few-shot setting. As shown in Table 8, CoRA consistently outperforms LIFT and C-LoRA, especially at lower sampling rates where LIFT and C-LoRA suffer from insufficient training data, leading to degraded performance. CoRA's superior performance is attributed to its comprehensive modeling of correlations and efficient utilization of TSFMs. As the sampling rate increases, LIFT and C-LoRA also improve TSFMs' performance, but at a higher sampling rate will be a higher training cost.

Table 8: The sampling rate set to {5%, 10%, 15%, 20%, 25%}. **Black**: the best, Underline: the 2nd best.

| Dataset | ETTm2 | | | | | Electricity | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Rate | 5% | 10% | 15% | 20% | 25% | 5% | 10% | 15% | 20% | 25% |
| TTM | 0.164 | 0.162 | 0.163 | 0.160 | 0.157 | 0.146 | 0.143 | 0.145 | 0.141 | 0.142 |
| + LIFT | 0.162 | 0.163 | 0.161 | 0.158 | 0.156 | 0.149 | 0.145 | 0.144 | 0.139 | 0.139 |
| + C-LoRA | 0.166 | 0.163 | 0.162 | 0.159 | 0.155 | 0.148 | 0.146 | 0.147 | 0.140 | 0.141 |
| + CoRA | **0.160**<br>±0.003 | **0.158**<br>±0.002 | **0.159**<br>±0.003 | **0.157**<br>±0.001 | **0.153**<br>±0.002 | **0.143**<br>±0.003 | **0.141**<br>±0.003 | **0.142**<br>±0.002 | **0.138**<br>±0.002 | **0.135**<br>±0.001 |

## F.2 Hyperparameter Sensitivity

With TTM as the backbone and H set to 96, we study the hyperparameter sensitivity of CoRA, including the Degree of Polynomial ($K$), the size of decomposition ($M$), the layers' number of projectors before and after HPCL ($N_1$ and $N_2$). Figure 6a show that $K$ is a robust hyperparameter, and we often choose 3 or 4 as common configurations. Figure 6b illustrates that the selection of M does not need to increase rapidly with the number of channels. Figure 6c and Figure 6d show that too few layers may lead to insufficient fitting capacity, while too many can diminish generalization ability. We often choose 3 or 5 as common configurations.

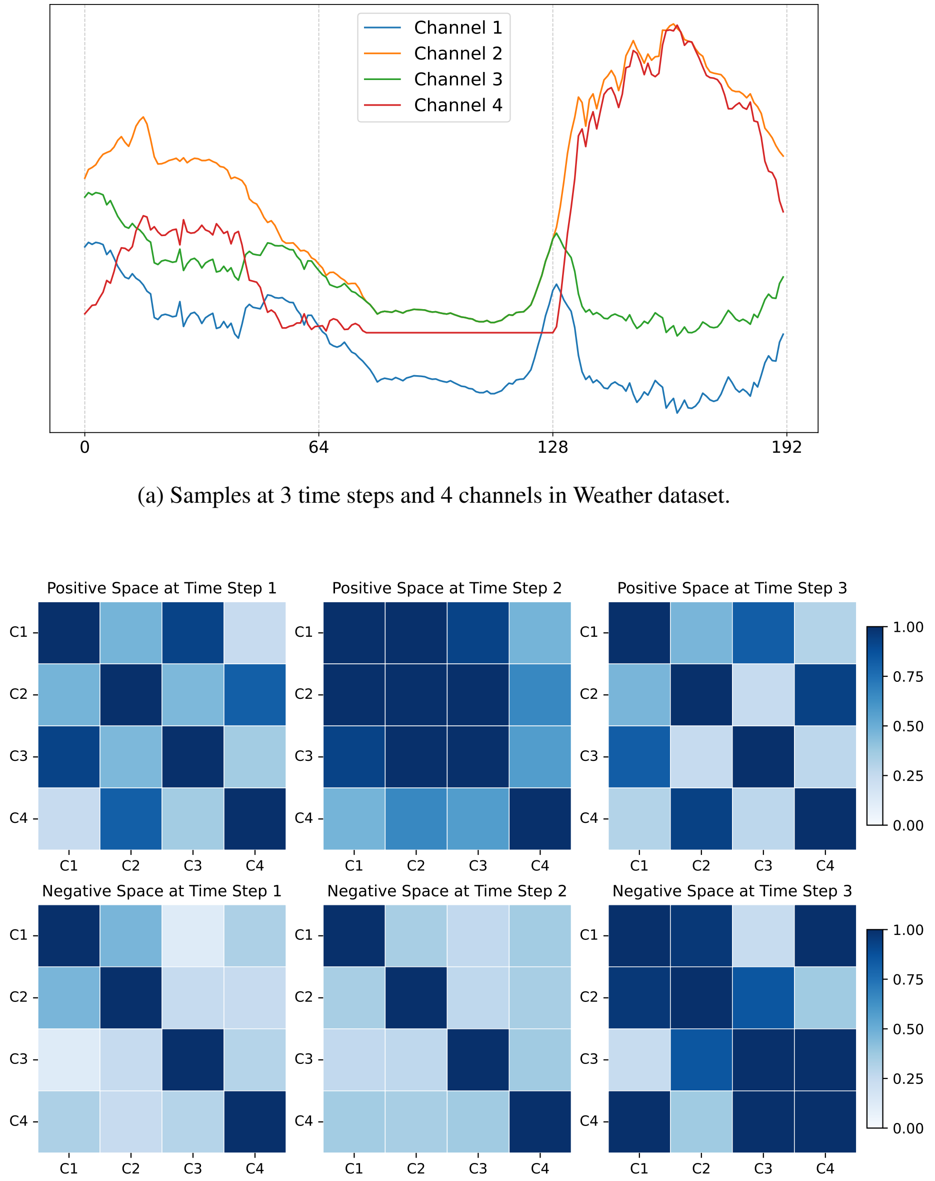


(a) Degree of Polynomial (b) Decomposition Size (c) $\mathcal{P}_1$,$\mathcal{P}_2$ Layers (d) $\mathcal{P}_3$,$\mathcal{P}_4$ Layers

Figure 6: Parameter sensitivity of main hyper-parameters in CoRA.

## F.3 Visualization of Heterogeneous Spaces.

To further demonstrate the effectiveness of CoRA in modeling the DCorr, HCorr and PCorr, we conduct a visualization experiment. Specifically, by examining samples at 3 time steps and 4 channels in the Weather dataset, we compare the similarities between representations in the heterogeneous space. The result is shown in Figure 7. Among them, Figure 7a illustrates the visualization of samples from the Weather dataset, with each time step comprising 64 time points. Figure 7b shows the cosine similarity between the channel representations in positive and negative spaces. Based on observations, it can be concluded that within the 0-64 time points, Channel 1 and Channel 3 as well as Channel 2 and Channel 4 exhibit significant positive correlations. Within the 64-128 time points,Channel 2 Channel 3 and Channel 4 show significant positive correlations, while Channel 3 demonstrates channel independence. Within the 128-192 time points,Channel 1 and Channel 3 exhibit significant positive correlations, while they show significant negative correlations with Channel 2 and Channel 4. These findings align with the actual data, demonstrating that CoRA is capable of simultaneously capturing DCorr HCorr and PCorr.



(a) Samples at 3 time steps and 4 channels in Weather dataset.

(b) The similarity of representations in positive and negative spaces at 3 time steps.

Figure 7: Visualization of Heterogeneous Spaces

### F.4 Comparison with channel-dependency TSFMs

To provide a more comprehensive comparison, we have conducted additional experiments with more channel-dependent TSFMs. The MSE results are summarized in the table below.

Table 9: The MSE results with channel-dependency TSFMs.

| Method | Moirai | Moirai+CoRA | UniTS | UniTS+CoRA | TTM | TTM+CoRA |
|---|---|---|---|---|---|---|
| ETT(Avg.) | 0.353±0.004 | 0.344±0.002 | 0.347±0.003 | 0.339±0.001 | 0.342±0.003 | 0.329±**0.002** |
| Weather | 0.257±0.003 | 0.241±0.003 | 0.235±0.002 | 0.220±0.002 | 0.226±0.003 | 0.214±**0.002** |
| AQShunyi | 0.690±0.003 | 0.672±0.002 | 0.717±0.002 | 0.685±0.002 | 0.701±0.003 | 0.678±**0.002** |
| ZafNoo | 0.519±0.003 | 0.497±0.001 | 0.508±0.004 | 0.491±0.002 | 0.505±0.003 | 0.483±**0.001** |

### F.5 Comparison with other plugins

Table 10: The MSE results of comparison.

| **Dataset** | **ETTm2** | | | **Weather** | | | **Electricity** | | |
|---|---|---|---|---|---|---|---|---|---|
| **Backbone** | **GPT4TS** | **Timer** | **UniTime** | **GPT4TS** | **Timer** | **UniTime** | **GPT4TS** | **Timer** | **UniTime** |
| **w/o Plugin** | 0.190 | 0.168 | 0.190 | 0.187 | 0.150 | 0.184 | 0.178 | 0.139 | 0.174 |
| **CoRA** | **0.183** | **0.164** | **0.183** | **0.180** | **0.146** | **0.174** | **0.176** | **0.136** | **0.170** |
| **LIFT** | 0.192 | 0.167 | 0.191 | 0.186 | 0.151 | 0.185 | 0.181 | 0.141 | 0.173 |
| **C-LoRA** | 0.199 | 0.171 | 0.199 | 0.190 | 0.155 | 0.182 | 0.182 | 0.142 | 0.178 |

### F.6 Evaluation on non-stationary datasets

To offer a dedicated analysis to investigate the model's behaviour in edge cases, we select four datasets with different Non-Stationary Rates (Qiu et al., 2024a) for evaluation. Our method combines both a rule-based and a learnable correlation matrix, which enhances its robustness. As shown in this result, our method still achieves a modest improvement even when the Non-Stationary Rate is as high as 0.360.

Table 11: MSE results for evaluation on non-stationary datasets.

| **Dataset** | **ETTh2** | **Weather** | **NASDAQ** | **Covid-19** |
|---|---|---|---|---|
| **Non-Stationary Rate** | 0.02 | 0.07 | 0.169 | 0.360 |
| **GPT4TS** | 0.377 | 0.254 | 1.411 | 1.972 |
| **+CoRA** | 0.361 | 0.243 | 1.387 | 1.924 |
| **Moment** | 0.369 | 0.251 | 1.208 | 2.356 |
| **+CoRA** | 0.356 | 0.243 | 1.174 | 2.307 |

### F.7 Evaluation on high-dimensional datasets

Table 12: MSE results for evaluation on high-dimensional datasets.

| **Dataset** | **Traffic** $N = 862$ | | **Covid-19** $N = 948$ | | **Wike2000** $N = 2000$ | |
|---|---|---|---|---|---|---|
| **Metric** | **MSE** | **Max-GPU Memroy** | **MSE** | **Max-GPU Memroy** | **MSE** | **Max-GPU Memroy** |
| **GPT4TS** | 0.441 | 7.73G | 1.972 | 12.53G | 547.024 | 23.96G |
| **+CoRA** | 0.430 | 8.95G | 1.924 | 14.24G | 535.811 | 28.44G |
| **Moment** | 0.453 | 5.37G | 2.356 | 6.21G | 525.352 | 11.12G |
| **+CoRA** | 0.437 | 7.02G | 2.307 | 7.89G | 517.165 | 17.41G |

We selected three high-dimensional datasets with more than 500 variables ($N > 500$). The MSE and maximum GPU memory usage for these datasets are reported in the table below. The results above show that even for a dataset with 2,000 variables, our method avoids introducing memory or numerical bottlenecks. Moreover, it is still capable of learning correlations to a certain degree, leading to performance enhancements for the TSFM.

### F.8 Evaluation on different tasks for time series

To explore the capabilities of CoRA on tasks beyond forecasting, we conducted relevant experiments. For anomaly detection, we use the **MSL** and **SMAP** as evaluation datasets (Qiu et al., 2025b). For classification, we select the **FaceDetection**, **Heartbeat**, and **PEMS-SF** as evaluation datasets (Goswami et al., 2024). For the anomaly detection task, we evaluate performance using the **VUS_ROC** and **VUS_PR** metrics. For the classification task, we use **Accuracy**. The results of all experiments are summarized in the Table 13.

Table 13: Evaluation on different tasks for time series.

| **Task** | **Anomaly detection** | | | | **Classification** | | |
|---|---|---|---|---|---|---|---|
| **Dataset** | **MSL** | | **SMAP** | | **FaceDetection** | **Heartbeat** | **PEMS-SF** |
| **Metric** | **VUS-ROC** | **VUS-PR** | **VUS-ROC** | **VUS-PR** | **Accuracy** | **Accuracy** | **Accuracy** |
| **GPT4TS** | 0.624 | 0.195 | 0.504 | 0.147 | 0.683 | 0.776 | 0.874 |
| **+CoRA** | 0.628 | 0.200 | 0.510 | 0.149 | 0.688 | 0.791 | 0.876 |
| **Moment** | 0.663 | 0.212 | 0.474 | 0.127 | 0.675 | 0.786 | 0.866 |
| **+CoRA** | 0.667 | 0.214 | 0.483 | 0.130 | 0.681 | 0.789 | 0.873 |

The results indicate that although CoRA was not specifically designed for these tasks, its direct application still yields performance improvements. This demonstrates CoRA's effectiveness in enhancing TSFMs by capturing correlation.

## G The Use of Large Language Models

The use of open-source Large Language Models (LLMs) in this work was strictly limited to assisting with the translation of certain terms and polishing a small portion of the text. LLMs did not contribute to the conceptual aspects of the research, including information retrieval, knowledge discovery, or the ideation process.