

From Similarity to Superiority: Channel Clustering for Time Series Forecasting

Jialin Chen¹ Jan Eric Lenssen^{2,3} Aosong Feng¹ Weihua Hu² Matthias Fey² Leandros Tassiulas¹
Jure Leskovec^{2,4} Rex Ying^{1,2}

Abstract

Time series forecasting has attracted significant attention in recent decades. Previous studies have demonstrated that the Channel-Independent (CI) strategy improves forecasting performance by treating different channels individually, while it leads to poor generalization on unseen instances and ignores potentially necessary interactions between channels. Conversely, the Channel-Dependent (CD) strategy mixes all channels with even irrelevant and indiscriminate information, which, however, results in oversmoothing issues and limits forecasting accuracy. There is a lack of channel strategy that effectively balances individual channel treatment for improved forecasting performance without overlooking essential interactions between channels. Motivated by our observation of a correlation between the time series model’s performance boost against channel mixing and the intrinsic similarity on a pair of channels, we developed a novel and adaptable Channel Clustering Module (CCM). CCM dynamically groups channels characterized by intrinsic similarities and leverages cluster identity instead of channel identity, combining the best of CD and CI worlds. Extensive experiments on real-world datasets demonstrate that CCM can (1) boost the performance of CI and CD models by an average margin of 2.4% and 7.2% on long-term and short-term forecasting, respectively; (2) enable zero-shot forecasting with mainstream time series forecasting models; (3) uncover intrinsic time series patterns among channels and improve interpretability of complex time series models.

1. Introduction

Time series forecasting has been attracting a surge of interest across diverse fields, ranging from economics, energy (Alvarez et al., 2010; Koprinska et al., 2018), weather (Angryk

¹Yale University ²Kumo.AI ³Max Planck Institute for Informatics ⁴Stanford University. Correspondence to: Rex Ying <rex.ying@yale.edu>. Preprint.

et al., 2020; Lam et al., 2023), to transportation planning (Li et al., 2015; Yin & Shang, 2016). The complexity of the task is heightened by factors including seasonality, trend, noise in the data, and potential cross-channel information.

Despite the numerous deep learning time series models proposed recently (Chen et al., 2023; Zeng et al., 2022; Wen et al., 2022; Zhou et al., 2022b; 2021; Wu et al., 2021; 2022; Liu et al., 2022a), an unresolved challenge persists in the effective management of channel interaction within the forecasting framework (Li et al., 2023a; Peiwen & Changsheng, 2023). Previous works have explored two primary channel strategies: **Channel-Independent (CI)** and **Channel-Dependent (CD)** strategies. Channel-Independent (CI) strategy has shown promise in better forecasting performance by having individual models for each channel. However, a critical drawback is its limited generalizability and robustness on unseen channels (Han et al., 2023). Besides, it tends to overlook potential interactions between various channels. Conversely, Channel-Dependent (CD) strategy models all channels as a whole and captures intricate channel relations, while they tend to show oversmoothing and have trouble fitting to individual channels, especially when the similarity between channels is very low. Moreover, existing models typically treat univariate data in a CI manner, neglecting the interconnections between time series samples, even though these dependencies are commonly observed in real-world scenarios, such as stock market or weather forecasting (Ni & Xu, 2023; Yin et al., 2021; Jang & Matteson, 2023).

Proposed work. To address the aforementioned challenges, we propose a Channel Clustering Module (CCM) that balances individual channel treatment and captures necessary cross-channel dependencies simultaneously. CCM is motivated by the following key observations: All models, CI and CD, heavily rely on channel identity information. The level of reliance is anti-correlated with the similarity between channels (see Sec. 4.1 for an analysis). This intriguing phenomenon alludes to the model’s analogous behavior on similar channels. The proposed CCM thereby involves the strategic clustering of channels into cohesive clusters, where intra-cluster channels exhibit a higher degree of similarity, effectively replacing channel identity with cluster identity. To capture the underlying time series patterns within these clusters, we employ **cluster-aware Feed Forward** to

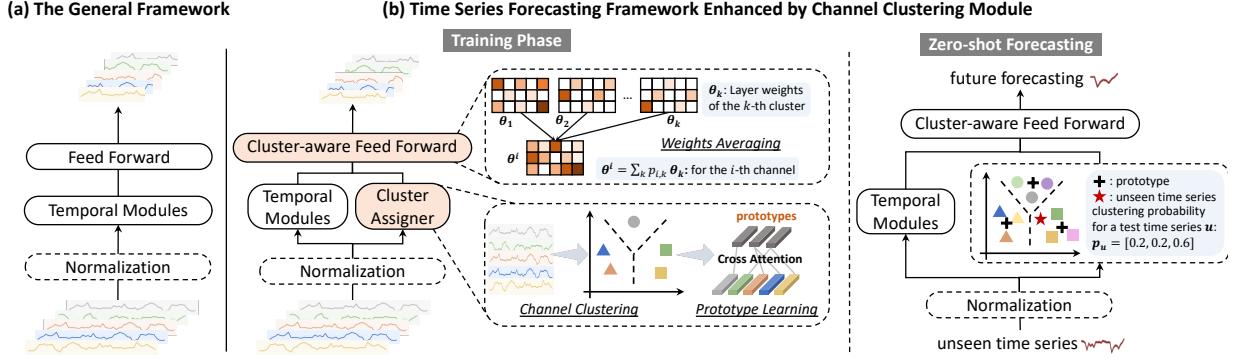


Figure 1. The pipeline of applying Channel Clustering Module (CCM) to general time series models. (a) illustrates the general framework of most time series models. (b) illustrates two core modules modified when applying CCM to base models: cluster assigner and cluster-aware Feed Forward. The cluster assigner learns channel clustering based on intrinsic similarities and creates prototype embeddings for each cluster via a cross-attention mechanism. The clustering probabilities $\{p_{i,k}\}$ are subsequently used in the cluster-aware Feed Forward to average $\{\theta_k\}_{k=1}^K$, which are layer weights assigned to K clusters, obtaining weights θ^i for the i -th channel. The learned prototypes retain pre-trained knowledge, enabling zero-shot forecasting on unseen samples in both univariate and multivariate scenarios.

ensure individual management of each cluster. Moreover, CCM learns expressive **prototype embeddings** in the training phase, which enable **zero-shot forecasting** on unseen samples by grouping them into appropriate clusters.

CCM is a plug-and-play solution that is adaptable to most mainstream time series models. We evaluate the effectiveness of CCM on four different time series backbones (*aka*, base models), TSMixer (Chen et al., 2023), DLinear (Zeng et al., 2022), PatchTST (Nie et al., 2022), and TimesNet (Wu et al., 2022). Extensive experiments verify the superiority of CCM in long-term and short-term forecasting benchmarks, achieving an average margin of 2.4% and 7.2%, respectively. Additionally, we collect stock data from a diverse range of companies to construct a new stock univariate dataset. Leveraging information from intra-cluster samples, CCM consistently shows a stronger ability to accurately forecast stock prices within the dynamic and intricate stock market. Moreover, CCM enhances zero-shot forecasting capacities of time series backbones in cross-domain scenarios, which further highlights the robustness and versatility of CCM.

The **contributions** of this paper are: (1) We propose a novel and unified channel strategy, *i.e.*, CCM, which is adaptable to most mainstream time series models. CCM explores the optimal trade-off between channel individual treatment and cross-channel modeling, (2) Through learning prototypes from clusters, CCM enables zero-shot forecasting on unseen samples in both univariate and multivariate scenarios, and (3) CCM demonstrates superiority in improving performance on long-term and short-term forecasting.

2. Related Work

2.1. Time Series Forecasting Models

Traditional machine learning methods such as Prophet (Taylor & Letham, 2018; Triebe et al., 2021), ARIMA (Zhang, 2003) capture the trend component and seasonality in time

series (Ahmed et al., 2010). As data availability continues to grow, deep learning methods revolutionized this field, introducing more complex and efficient models (Torres et al., 2021; Lim & Zohren, 2021). Convolutional Neural Networks (CNNs) (Wu et al., 2022; Liu et al., 2022a; Bai et al., 2018; Wan et al., 2019; Sen et al., 2019), have been widely adopted to capture local temporal dependencies. Recurrent Neural Networks (RNNs) (Hewamalage et al., 2021; Rangapuram et al., 2018; Smyl, 2020; Salinas et al., 2020; Bai et al., 2018) excel in capturing sequential information, yet they often struggle with longer sequences. Transformer-based models (Zhou et al., 2021; Liu et al., 2022b; Wu et al., 2021; Nie et al., 2022; Zhang & Yan, 2022; Liu et al., 2023b; Wen et al., 2022; Tang & Matteson, 2021; Zhou et al., 2022b; Liu et al., 2021; Feng et al., 2024), typically equipped with self-attention mechanisms (Vaswani et al., 2017), demonstrate their proficiency in handling long-range dependencies, although they require substantial computational resources. Recently, linear models (Oreshkin et al., 2019; Zhang et al., 2022; Das et al., 2023), *e.g.*, DLinear (Zeng et al., 2022), TSMixer (Chen et al., 2023), have gained popularity for their simplicity and effectiveness in long-term time series forecasting, but they may underperform with non-linear and complex patterns. Besides, traditional tricks, including trend-seasonal decomposition (Zeng et al., 2022; Cleveland et al., 1990; Wen et al., 2020) and multi-periodicity analysis (Benaouda et al., 2006; Percival & Walden, 2000; Wu et al., 2022; Wang et al., 2022; Zhang et al., 2024; Yi et al., 2023; Zhou et al., 2022a) continue to play a crucial role in aiding the preprocessing stage for advanced models.

2.2. Channel Strategies in Time Series Forecasting

Most deep learning models (Wu et al., 2021; Liu et al., 2021; Zhou et al., 2022b) adopt the Channel-Dependent (CD) strategy, aiming to harness the full spectrum of information across channels. Conversely, the Channel-Independent (CI)

approaches (Nie et al., 2022; Zeng et al., 2022) build forecasting models for each channel independently. Prior works on CI and CD strategy (Han et al., 2023; Li et al., 2023a; Montero-Manso & Hyndman, 2021) present that CI has high capacity and low robustness, whereas CD is the opposite. Predicting residuals with regularization (PRReg) (Han et al., 2023) is thereby proposed to incorporate a regularization term in the objective to encourage smoothness in future forecasting. However, the essential challenge of channel strategies from the model design perspective has not been solved and it remains challenging to develop a balanced channel strategy with both high capacity and robustness.

Prior research has explored effective clustering of channels to improve the predictive capabilities in diverse applications, including image classification (Jiang et al., 2010), natural language processing (NLP) (Marin et al., 2023; George & Sumathy, 2023), anomaly detection (Li et al., 2012; Syarif et al., 2012; Gunupudi et al., 2017). For instance, in traffic prediction (Ji et al., 2023; Liu et al., 2023a), clustering techniques have been proposed to group related traffic regions to capture intricate spatial patterns. Despite the considerable progress in these areas, the potential and effect of channel clustering in time series forecasting remain under-explored.

3. Preliminaries

Time Series Forecasting. Formally, let $X = [x_1, \dots, x_T] \in \mathbb{R}^{T \times C}$ be a time series, where T is the length of historical data. $x_t \in \mathbb{R}^C$ represents the observation at time t . C denotes the number of variates (*i.e.*, channels). In the univariate case (*i.e.*, $C = 1$), the series is composed of single observations at each time step, whereas in the multivariate case (*i.e.*, $C > 1$), each x_t is a vector of multiple observations. The objective is to construct a predictive model f that estimates the future values of the series, $Y = [\hat{x}_{T+1}, \dots, \hat{x}_{T+H}] \in \mathbb{R}^{H \times C}$, where H is the forecasting horizon. We use $X_{[:,i]} \in \mathbb{R}^T$ (X_i for simplicity) to denote the i -th channel in the time series.

Channel Dependent (CD) and Channel Independent (CI). The CI strategy models each channel X_i separately and ignores the potential channel interactions. This approach is typically denoted as $f^{(i)} : \mathbb{R}^T \rightarrow \mathbb{R}^H$ for $i = 1, \dots, C$, where $f^{(i)}$ gets access to the historical data specific to the i -th channel. In contrast, the CD strategy models all the channels as a whole, using a function $f : \mathbb{R}^{T \times C} \rightarrow \mathbb{R}^{H \times C}$. This strategy is essential in scenarios where channels are not just parallel data streams but are interrelated, such as in financial markets or traffic flows.

4. Proposed Method

In this work, we propose Channel Clustering Module (CCM), a model-agnostic method that is adaptable to most

mainstream models. The pipeline of applying CCM is visualized in Figure 1. General time series models, shown in Figure 1(a), typically consist of three core components (Li et al., 2023a;b): an optional **normalization layer** (*e.g.*, RevIN (Kim et al., 2021), SAN (Liu et al., 2023c)), **temporal modules** including linear layers, transformer-based, or convolutional backbones, and a **feed-forward layer** that forecasts the future values. Motivated by the empirical observation discussed in Sec. 4.1, CCM presents with a cluster assigner preceding the temporal modules, followed by a cluster-aware Feed Forward (Sec. 4.2). The cluster assigner implements channel clustering based on intrinsic similarities and employs a cross-attention mechanism to generate prototypes for each cluster, which stores the knowledge from the training set and endows the model with zero-shot forecasting capacities.

4.1. Motivation for Channel Similarity

To motivate our similarity-based clustering method, we provide a toy experiment to verify the following two assumptions: **(1)** Existing forecasting methods heavily rely on channel identity information. **(2)** This reliance clearly anti-correlates with channel similarity: for channels with high similarity, channel identity information is less important. Together, these two assumptions motivate us to design an approach that provides cluster identity instead of channel identity, combining the best of both worlds: high capacity and generalizability.

Table 1. Averaged performance gain from a channel identity information ($\Delta\mathcal{L}(\%)$) and Pearson Correlation Coefficients (PCC) between $\{\Delta\mathcal{L}_{ij}\}_{i,j}$ and $\{\text{SIM}(X_i, X_j)\}_{i,j}$. The values are averaged across all test samples.

| Base Model | TSMixer | DLinear | PatchTST | TimesNet | |
|------------------|-------------------------|---------|----------|----------|-------|
| Channel Strategy | CD | CI | CI | CD | |
| ETTh1 | $\Delta\mathcal{L}(\%)$ | 2.67 | 1.10 | 11.30 | 18.90 |
| | PCC | -0.67 | -0.66 | -0.61 | -0.66 |
| ETTm1 | $\Delta\mathcal{L}(\%)$ | 4.41 | 5.55 | 6.83 | 14.98 |
| | PCC | -0.68 | -0.67 | -0.68 | -0.67 |
| Exchange | $\Delta\mathcal{L}(\%)$ | 16.43 | 19.34 | 27.98 | 24.57 |
| | PCC | -0.62 | -0.62 | -0.47 | -0.49 |

We select four recent and popular time series models with different backbones. TSMixer (Chen et al., 2023) and DLinear are linear models. PatchTST (Nie et al., 2022) is a transformer-based model with a patching mechanism and TimesNet (Wu et al., 2022) is a convolutional network that captures multi-periodicity in data. Among these, TSMixer and TimesNet utilize a Channel-Dependent strategy while DLinear and PatchTST adopt the Channel-Independent design. We train a time series model across all channels and evaluate the channel-wise Mean Squared Error (MSE) loss on the test set. Then, we repeat training while randomly shuffling channels in each batch. Note that for both CD and CI models, this means channel identity information will be removed. We report the average performance gain across all channels based on the random shuffling experiments (de-

noted as $\Delta\mathcal{L}(\%)$) in Table 1. We see that all models rely on channel identity information to achieve better performance.

Next, we show that this performance gap anti-correlates with channel similarity. We define channel similarity based on radial basis function kernels (Zhang et al., 2004) as

$$\text{SIM}(X_i, X_j) = \exp\left(\frac{-\|X_i - X_j\|^2}{2\sigma^2}\right), \quad (1)$$

where σ is a scaling factor. Note that the similarity is computed on the standardized time series to avoid scaling differences. More details are discussed in Appendix A. The performance difference in MSE from the random shuffling experiment for channel i is denoted as $\Delta\mathcal{L}_i$. We define $\Delta\mathcal{L}_{ij} := |\Delta\mathcal{L}_i - \Delta\mathcal{L}_j|$ and calculate the Pearson Correlation Coefficients between $\{\Delta\mathcal{L}_{ij}\}_{i,j}$ and $\{\text{SIM}(X_i, X_j)\}_{i,j}$, as shown in Table 1, consistently verifying our assumption.

4.2. CCM: Channel Clustering Module

Channel Clustering. Motivated by the above observations, we first initialize a set of K cluster embedding $\{c_1, \dots, c_K\}$, where $c_k \in \mathbb{R}^d$, d is the hidden dimension and K is a hyperparameter. Given a multivariate time series $X \in \mathbb{R}^{T \times C}$, each channel in the input X_i is transformed into a d -dimensional channel embedding h_i through an MLP. The probability that a given channel X_i is associated with the k -th cluster is the normalized inner-product of the cluster embedding c_k and the channel embedding h_i , which is computed as

$$p_{i,k} = \text{Normalize}\left(\frac{c_k^\top h_i}{\|c_k\| \|h_i\|}\right) \in [0, 1]. \quad (2)$$

The normalization operator ensures that $\sum_k p_{i,k} = 1$ and validates the clustering probability distribution across k clusters. We utilize reparameterization trick (Jang et al., 2016) to obtain the clustering membership matrix $M \in \mathbb{R}^{C \times K}$ where $M_{ik} \approx \text{Bernoulli}(p_{i,k})$. Higher probability $p_{i,k}$ results in M_{ik} close to 1, leading to deterministic existence of certain channel in the corresponding cluster.

Prototype Learning. The cluster assigner also creates a d -dimensional prototype embedding for each cluster in the training phase. Let $C = [c_1, \dots, c_K] \in \mathbb{R}^{K \times d}$ denote the cluster embedding, and $H = [h_1, \dots, h_C] \in \mathbb{R}^{C \times d}$ denote the hidden embedding of the channels. To emphasize the intra-cluster channels and remove interference from out-of-cluster channel information, we design a modified cross-attention as follows,

$$\hat{C} = \text{Normalize}\left(\exp\left(\frac{(W_Q C)(W_K H)^\top}{\sqrt{d}}\right) \odot M^\top\right) W_V H, \quad (3)$$

where the clustering membership matrix M is an approximately binary matrix to enable sparse attention on intra-cluster channels specifically. W_Q , W_K and W_V are learnable parameters. The prototype embedding $\hat{C} \in \mathbb{R}^{K \times d}$

serves as the updated cluster embedding for subsequent clustering probability computing in Eq. 2.

Cluster Loss. We further introduce a specifically designed loss function for the clustering quality, termed ClusterLoss, which incorporates both the alignment of channels with their respective clusters and the distinctness between different clusters in a self-supervised learning context. Let $S \in \mathbb{R}^{C \times C}$ denote the channel similarity matrix with $S_{ij} = \text{SIM}(X_i, X_j)$ defined in Eq. 1. The ClusterLoss is formulated as:

$$\mathcal{L}_C = -\text{Tr}(M^\top S M) + \text{Tr}((I - M M^\top) S), \quad (4)$$

where Tr indicates a trace operator. $\text{Tr}(M^\top S M)$ maximizes the channel similarities within clusters, which is a fundamental requirement for effective clustering. $\text{Tr}((I - M M^\top) S)$ instead encourages separation between clusters, which further prevents overlap and ambiguity in clustering assignments. \mathcal{L}_C captures meaningful time series prototypes without relying on external labels or annotations. The overall loss function thereby becomes $\mathcal{L} = \mathcal{L}_F + \beta \mathcal{L}_C$, where \mathcal{L}_F is the general forecasting loss such as MSE loss; and β is a regularization parameter for a balance between forecasting accuracy and cluster quality.

Cluster-aware Feed Forward. Instead of using individual Feed Forward per channel in a CI manner or sharing one Feed Forward across all channels in a CD manner, we assign a separate Feed Forward to each cluster to capture the underlying shared time series patterns within the clusters. In this way, we use cluster identity to replace channel identity. Each Feed Forward is parameterized with a single linear layer due to its efficacy in time series forecasting (Zeng et al., 2022; Li et al., 2023a; Chen et al., 2023). Let $h_{\theta_k}(\cdot)$ represent the linear layer for the k -th cluster with weights θ_k . Z_i represents the hidden embedding of the i -th channel before the last layer. The final forecast is thereby averaged across the outputs of all cluster-aware Feed Forward with $\{p_{i,k}\}$ as weights, e.g., $Y_i = \sum_k p_{i,k} h_{\theta_k}(Z_i)$ for the i -th channel. For computational efficiency, it is equivalent to $Y_i = h_{\theta^i}(Z_i)$ with averaged weights $\theta^i = \sum_k p_{i,k} \theta_k$.

Univariate Adaptation. In the context of univariate time series forecasting, we extend the proposed method to clustering on samples. We leverage the similarity between two univariate time series as defined in Eq. 1, and classify univariate time series with comparable patterns into the same cluster. This univariate adaptation allows it to capture interrelation within samples and extract valuable insights from analogous time series. This becomes particularly valuable in situations where meaningful dependencies exist among various univariate samples, such as the stock market.

Zero-shot Forecasting. The prototype embeddings acquired during the training phase serve as a compact representation of the pre-trained knowledge and can be harnessed

for seamless knowledge transfer to unseen samples or new channels. We apply this knowledge to unseen instances by computing the clustering probability distribution on the pre-trained clusters, following Eq. 2. The clustering probability is subsequently used for averaging cluster-aware Feed Forward. We disable the cross-attention operation to fix the prototype embeddings in zero-shot forecasting scenarios. It is worth noting that zero-shot forecasting is adaptable to both univariate and multivariate scenarios. We refer to Appendix B for detailed discussion.

4.3. Complexity Analysis

CCM effectively strikes a balance between the CI and CD strategies. On originally CI models, CCM introduces strategic clustering on channels, which not only reduces the model complexity but also enhances their generalizability. Simultaneously, CCM increases the model complexity on originally CD models with negligible overhead for higher capacities. We refer to Figure 5 in Appendix C.5 for empirical analysis. Theoretically, the computational complexity of clustering probability computation (Eq. 2) and the cross-attention (Eq. 3) are $\mathcal{O}(KCd)$, where K, C are the number of clusters and channels, respectively, and d is the hidden dimension. One may also use more attention mechanisms (Kitaev et al., 2020; Wang et al., 2020; Shen et al., 2021) for efficiency. The complexity of cluster-aware Feed Forward scales linearly in C, K , and the forecasting horizon H .

5. Experiments

CCM consistently improves performance based on CI or CD models by significant margins across multiple benchmarks and settings, including long-term forecasting on 9 public multivariate datasets (Sec. 5.2); short-term forecasting on 2 univariate datasets (Sec. 5.3); and zero-shot forecasting in cross-domain and cross-granularity scenarios (Sec. 5.4).

5.1. Experimental Setup

Datasets. For long-term forecasting, we experiment on 9 popular benchmarking datasets across diverse domains (Zhou et al., 2021; Wu et al., 2021; Lai et al., 2018), including weather, traffic and electricity. M4 dataset (Makridakis et al., 2018) is used in short-term forecasting, which is a univariate dataset that covers time series across diverse domains and various sampling frequencies from hourly to yearly. We further provide a new stock time series dataset with 1390 univariate time series. Each time series records the price history of an individual stock spanning 10 years. Due to the potential significant fluctuations in stock performance across different companies, this dataset poses challenges for models capturing diverse and evolving stock patterns in financial markets. The statistics of long-term and

short-term datasets are shown in Table 2 and Table 3.

Table 2. The statistics of datasets in long-term forecasting. Horizon is {96, 192, 336, 720}.

| Dataset | Channels | Length | Frequency |
|-------------|----------|--------|-----------|
| ETTh1&ETTh2 | 7 | 17420 | 1 hour |
| ETTm1&ETTm2 | 7 | 69680 | 15 min |
| ILI | 7 | 966 | 1 week |
| Exchange | 8 | 7588 | 1 day |
| Weather | 21 | 52696 | 10 min |
| Electricity | 321 | 26304 | 1 hour |
| Traffic | 862 | 17544 | 1 hour |

Table 3. Dataset details of M4 and Stock in short-term forecasting.

| Dataset | Length | Horizon |
|--------------|--------|---------|
| M4 Yearly | 23000 | 6 |
| M4 Quarterly | 24000 | 8 |
| M4 Monthly | 48000 | 18 |
| M4 Weekly | 359 | 13 |
| M4 Daily | 4227 | 14 |
| M4 Hourly | 414 | 48 |
| Stock (Ours) | 10000 | 7/24 |

We follow standard protocols (Zhou et al., 2021; Wu et al., 2021; 2022) for data splitting on public benchmarking datasets. As for the stock dataset, we divide the set of stocks into train/validation/test sets with a ratio of 7:2:1. Therefore, validation/test sets present unseen samples (*i.e.*, stocks) for model evaluation. This evaluation setting emphasizes the data efficiency aspect of time series models for scenarios where historical data is limited or insufficient for retraining from scratch given unseen instances. More details on datasets are provided in Appendix C.1.

Base Models and Experimental Details. CCM is a model-agnostic channel strategy that can be applied to arbitrary time series forecasting models for improved performance. We meticulously select four recent state-of-the-art time series models as base models: TSMixer (Chen et al., 2023), DLinear (Zeng et al., 2022), PatchTST (Nie et al., 2022) and TimesNet (Wu et al., 2022), which mainly cover three mainstream paradigms, including linear models, transformer-based and convolutional models. For fair evaluation, we use the optimal experiment configuration as provided in the official code to implement both base models and the enhanced version with CCM. All the experiments are implemented by PyTorch on a single NVIDIA RTX A6000 48GB GPU. Experiment configurations and implementations are detailed in Appendix C.3. Experimental results in the following sections are averaged on five runs with different random seeds. Refer to Appendix C.6 for standard deviation results.

5.2. Long-term Forecasting Results

We report the mean squared error (MSE) and mean absolute error (MAE) on nine real-world datasets for long-term forecasting evaluation in Table 2. The forecasting horizon is {96, 192, 336, 720}. From the table, we observe that the model enhanced with CCM outperforms the base model in general. Specifically, CCM improves the long-term forecasting performance in 90.27% cases in MSE and 84.03% cases in MAE across 144 different experiment settings. Remarkably, CCM achieves a substantial boost on DLinear, with a significant reduction on MSE by 5.12% and MAE by 3.04%. The last column of the table quantifies the average percentage improvement in terms of MSE/MAE, which underscores the consistent enhancement brought by CCM across all forecasting horizons and datasets.

Table 4. Long-term forecasting results on 9 real-world datasets in terms of MSE and MAE, the lower the better. The forecasting horizons are {96, 192, 336, 720}. The better performance in each setting is shown in **bold**. The best results for each row are underlined. The last column shows the average percentage of MSE/MAE improvement of CCM over four base models.

| Model Metric | TSMixer | | | | + CCM | | DLLinear | | | | + CCM | | PatchTST | | + CCM | | TimesNet | | + CCM | | IMP(%) |
|--------------|---------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|-------|-------|-----|--------|
| | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | |
| ETTh1 | 96 | 0.361 | 0.392 | 0.365 | 0.393 | 0.375 | 0.399 | 0.371 | 0.393 | 0.375 | 0.398 | 0.371 | 0.396 | 0.384 | 0.402 | 0.380 | 0.400 | 0.539 | | | |
| | 192 | 0.404 | 0.418 | 0.402 | 0.418 | 0.405 | 0.416 | 0.404 | 0.415 | 0.415 | 0.425 | 0.414 | 0.424 | 0.436 | 0.429 | 0.431 | 0.425 | 0.442 | | | |
| | 336 | 0.422 | 0.430 | 0.423 | 0.430 | 0.445 | 0.440 | 0.438 | 0.443 | 0.422 | 0.440 | 0.417 | 0.429 | 0.491 | 0.469 | 0.485 | 0.461 | 0.908 | | | |
| | 720 | 0.463 | 0.472 | 0.462 | 0.470 | 0.489 | 0.479 | 0.497 | 0.449 | 0.468 | 0.447 | 0.469 | 0.521 | 0.500 | 0.520 | 0.493 | 0.333 | | | | |
| ETTm1 | 96 | 0.285 | 0.339 | 0.283 | 0.337 | 0.299 | 0.343 | 0.298 | 0.343 | 0.294 | 0.351 | 0.289 | 0.338 | 0.338 | 0.375 | 0.335 | 0.371 | 1.123 | | | |
| | 192 | 0.339 | 0.365 | 0.336 | 0.368 | 0.335 | 0.365 | 0.334 | 0.365 | 0.334 | 0.370 | 0.333 | 0.363 | 0.374 | 0.387 | 0.373 | 0.383 | 0.482 | | | |
| | 336 | 0.361 | 0.406 | 0.359 | 0.393 | 0.370 | 0.386 | 0.365 | 0.385 | 0.373 | 0.397 | 0.370 | 0.392 | 0.410 | 0.411 | 0.412 | 0.416 | 0.716 | | | |
| | 720 | 0.445 | 0.470 | 0.424 | 0.421 | 0.427 | 0.423 | 0.424 | 0.417 | 0.416 | 0.420 | 0.419 | 0.430 | 0.478 | 0.450 | 0.477 | 0.448 | 1.852 | | | |
| ETTh2 | 96 | 0.284 | 0.343 | 0.278 | 0.338 | 0.289 | 0.353 | 0.285 | 0.348 | 0.278 | 0.340 | 0.274 | 0.336 | 0.340 | 0.374 | 0.336 | 0.371 | 1.371 | | | |
| | 192 | 0.339 | 0.385 | 0.325 | 0.393 | 0.384 | 0.418 | 0.376 | 0.413 | 0.341 | 0.382 | 0.339 | 0.355 | 0.402 | 0.414 | 0.400 | 0.410 | 1.806 | | | |
| | 336 | 0.361 | 0.406 | 0.361 | 0.399 | 0.442 | 0.459 | 0.438 | 0.455 | 0.329 | 0.384 | 0.327 | 0.383 | 0.452 | 0.452 | 0.449 | 0.445 | 0.823 | | | |
| | 720 | 0.445 | 0.470 | 0.438 | 0.464 | 0.601 | 0.549 | 0.499 | 0.496 | 0.381 | 0.424 | 0.378 | 0.415 | 0.462 | 0.468 | 0.457 | 0.461 | 4.370 | | | |
| ETTm2 | 96 | 0.171 | 0.260 | 0.167 | 0.260 | 0.167 | 0.260 | 0.166 | 0.258 | 0.174 | 0.261 | 0.168 | 0.256 | 0.187 | 0.267 | 0.189 | 0.270 | 0.860 | | | |
| | 192 | 0.221 | 0.296 | 0.220 | 0.296 | 0.284 | 0.352 | 0.243 | 0.323 | 0.238 | 0.307 | 0.231 | 0.300 | 0.249 | 0.309 | 0.250 | 0.310 | 3.453 | | | |
| | 336 | 0.276 | 0.329 | 0.277 | 0.330 | 0.369 | 0.427 | 0.295 | 0.358 | 0.293 | 0.346 | 0.275 | 0.331 | 0.321 | 0.351 | 0.318 | 0.347 | 6.012 | | | |
| | 720 | 0.420 | 0.422 | 0.369 | 0.391 | 0.554 | 0.522 | 0.451 | 0.456 | 0.373 | 0.401 | 0.374 | 0.400 | 0.408 | 0.403 | 0.394 | 0.391 | 7.139 | | | |
| Exchange | 96 | 0.089 | 0.209 | 0.085 | 0.206 | 0.088 | 0.215 | 0.085 | 0.214 | 0.094 | 0.216 | 0.088 | 0.208 | 0.107 | 0.234 | 0.105 | 0.231 | 2.880 | | | |
| | 192 | 0.195 | 0.315 | 0.177 | 0.300 | 0.178 | 0.317 | 0.171 | 0.306 | 0.191 | 0.311 | 0.185 | 0.309 | 0.226 | 0.344 | 0.224 | 0.340 | 3.403 | | | |
| | 336 | 0.343 | 0.421 | 0.312 | 0.405 | 0.371 | 0.462 | 0.300 | 0.412 | 0.343 | 0.427 | 0.342 | 0.423 | 0.367 | 0.448 | 0.361 | 0.442 | 5.875 | | | |
| | 720 | 0.898 | 0.710 | 0.847 | 0.697 | 0.966 | 0.754 | 0.811 | 0.683 | 0.888 | 0.706 | 0.813 | 0.673 | 0.964 | 0.746 | 0.957 | 0.739 | 5.970 | | | |
| ILI | 24 | 1.914 | 0.879 | 1.938 | 0.874 | 2.215 | 1.081 | 1.935 | 0.935 | 1.593 | 0.757 | 1.561 | 0.750 | 2.317 | 0.934 | 2.139 | 0.936 | 4.483 | | | |
| | 36 | 1.808 | 0.858 | 1.800 | 0.851 | 2.142 | 0.977 | 1.938 | 0.942 | 1.768 | 0.794 | 1.706 | 0.780 | 1.972 | 0.920 | 1.968 | 0.914 | 2.561 | | | |
| | 48 | 1.797 | 0.873 | 1.796 | 0.867 | 2.335 | 1.056 | 2.221 | 1.030 | 1.799 | 0.916 | 1.774 | 0.892 | 2.238 | 0.940 | 2.229 | 0.937 | 1.602 | | | |
| | 60 | 1.859 | 0.895 | 1.810 | 0.876 | 2.479 | 1.088 | 2.382 | 1.096 | 1.850 | 0.943 | 1.735 | 0.880 | 2.027 | 0.928 | 2.041 | 0.930 | 2.491 | | | |
| Weather | 96 | 0.149 | 0.198 | 0.147 | 0.194 | 0.192 | 0.250 | 0.187 | 0.245 | 0.149 | 0.198 | 0.147 | 0.197 | 0.172 | 0.220 | 0.169 | 0.215 | 1.729 | | | |
| | 192 | 0.201 | 0.248 | 0.192 | 0.242 | 0.248 | 0.297 | 0.240 | 0.285 | 0.194 | 0.241 | 0.191 | 0.238 | 0.219 | 0.261 | 0.215 | 0.257 | 2.539 | | | |
| | 336 | 0.264 | 0.291 | 0.244 | 0.281 | 0.284 | 0.335 | 0.274 | 0.324 | 0.244 | 0.282 | 0.245 | 0.285 | 0.280 | 0.306 | 0.274 | 0.291 | 2.924 | | | |
| | 720 | 0.320 | 0.336 | 0.318 | 0.334 | 0.339 | 0.374 | 0.320 | 0.357 | 0.320 | 0.334 | 0.316 | 0.333 | 0.365 | 0.359 | 0.366 | 0.362 | 1.476 | | | |
| Electricity | 96 | 0.142 | 0.237 | 0.139 | 0.235 | 0.153 | 0.239 | 0.142 | 0.247 | 0.138 | 0.233 | 0.136 | 0.231 | 0.168 | 0.272 | 0.158 | 0.259 | 2.480 | | | |
| | 192 | 0.154 | 0.248 | 0.147 | 0.246 | 0.158 | 0.251 | 0.152 | 0.248 | 0.153 | 0.247 | 0.153 | 0.247 | 0.184 | 0.289 | 0.172 | 0.262 | 3.226 | | | |
| | 336 | 0.163 | 0.264 | 0.161 | 0.262 | 0.170 | 0.269 | 0.168 | 0.267 | 0.170 | 0.263 | 0.168 | 0.262 | 0.198 | 0.300 | 0.181 | 0.284 | 2.423 | | | |
| | 720 | 0.208 | 0.300 | 0.204 | 0.299 | 0.233 | 0.342 | 0.230 | 0.338 | 0.206 | 0.296 | 0.210 | 0.301 | 0.220 | 0.320 | 0.205 | 0.309 | 1.417 | | | |
| Traffic | 96 | 0.376 | 0.264 | 0.375 | 0.262 | 0.411 | 0.284 | 0.411 | 0.282 | 0.360 | 0.249 | 0.357 | 0.246 | 0.593 | 0.321 | 0.554 | 0.316 | 1.488 | | | |
| | 192 | 0.397 | 0.277 | 0.340 | 0.279 | 0.423 | 0.287 | 0.422 | 0.286 | 0.379 | 0.256 | 0.379 | 0.254 | 0.617 | 0.336 | 0.562 | 0.331 | 3.175 | | | |
| | 336 | 0.413 | 0.290 | 0.411 | 0.289 | 0.438 | 0.299 | 0.436 | 0.297 | 0.401 | 0.270 | 0.389 | 0.255 | 0.629 | 0.336 | 0.579 | 0.341 | 2.120 | | | |
| | 720 | 0.444 | 0.306 | 0.441 | 0.302 | 0.467 | 0.316 | 0.471 | 0.318 | 0.443 | 0.294 | 0.430 | 0.281 | 0.640 | 0.350 | 0.587 | 0.366 | 1.445 | | | |

Comparison with Regularization Solution. PRReg (Han et al., 2023) recently displays to improve the performance of CD models, by predicting residuals with a regularization term in the training objective. We evaluate the effectiveness of PRReg and our proposed CCM on long-term forecasting performance enhancement based on CI and CD models. Following the previous training setting (Han et al., 2023), we develop CI and CD versions for Linear (Zeng et al., 2022) and Transformer (Vaswani et al., 2017) and report MSE loss. Table 5 shows that CCM surpasses PRReg in most cases, highlighting its efficacy compared with regularization solutions. See full results in Appendix C.4.

5.3. Short-term Forecasting Results

In both M4 and stock datasets, we follow the univariate forecasting setting. For M4 benchmarks, we adopt the evaluation setting in prior works (Oreshkin et al., 2019) and report the symmetric mean absolute percentage error (SMAPE), mean absolute scaled error (MASE), and overall weighted average (OWA) in Table 6. As for the stock dataset, we implement MAE and MSE as metrics in Table 7. See more

Table 5. Comparison between CCM and existing regularization method for improved performance on CI/CD strategies. The best results are highlighted in **bold**. The forecasting horizon is 24 for ILI dataset and 48 for other datasets. * denotes our implementation. Other results collect from (Han et al., 2023)

| | | CD | CI | +PRReg | +CCM* |
|-------------|-------------|--------------|-------|--------------|--------------|
| ETTh1 | Linear | 0.402 | 0.345 | 0.342 | 0.342 |
| | Transformer | 0.861 | 0.655 | 0.539 | 0.518 |
| ETTm1 | Linear | 0.404 | 0.354 | 0.311 | 0.310 |
| | Transformer | 0.458 | 0.379 | 0.349 | 0.300 |
| Weather | Linear | 0.142 | 0.169 | 0.131 | 0.130 |
| | Transformer | 0.251 | 0.168 | 0.180 | 0.164 |
| ILI | Linear | 2.343 | 2.847 | 2.299 | 2.279 |
| | Transformer | 5.309 | 4.307 | 3.254 | 3.206 |
| Electricity | Linear | 0.195 | 0.196 | 0.196 | 0.195 |
| | Transformer | 0.250 | 0.185 | 0.185 | 0.183 |

details on metrics in Appendix C.2.

Remarkably, the efficacy of CCM is consistent across all M4 sub-datasets with different sampling frequencies. Specifically, CCM outperforms the state-of-the-art linear model (DLlinear) by a significant margin of 11.62%, and outperforms the best convolutional method TimesNet by 8.88%.

Table 6. Short-term forecasting results on M4 dataset in terms of SMAPE, MASE, and OWA, the lower the better. The better performance in each setting is shown in **bold**.

| Model | | TSMixer | + CCM | DLinear | + CCM | PatchTST | + CCM | TimesNet | + CCM | IMP(%) |
|-----------|-------|--------------|---------------|---------|---------------|----------|---------------|----------|---------------|--------|
| Yearly | SMAPE | 14.702 | 14.676 | 16.965 | 14.337 | 13.477 | 13.304 | 15.378 | 14.426 | 7.286 |
| | MASE | 3.343 | 3.370 | 4.283 | 3.144 | 3.019 | 2.997 | 3.554 | 3.448 | 9.589 |
| | OWA | 0.875 | 0.873 | 1.058 | 0.834 | 0.792 | 0.781 | 0.918 | 0.802 | 11.346 |
| Quarterly | SMAPE | 11.187 | 10.989 | 12.145 | 10.513 | 10.380 | 10.359 | 10.465 | 10.121 | 6.165 |
| | MASE | 1.346 | 1.332 | 1.520 | 1.243 | 1.233 | 1.224 | 1.227 | 1.183 | 7.617 |
| | OWA | 0.998 | 0.984 | 1.106 | 0.931 | 0.921 | 0.915 | 0.923 | 0.897 | 6.681 |
| Monthly | SMAPE | 13.433 | 13.407 | 13.514 | 13.370 | 12.959 | 12.672 | 13.513 | 12.790 | 2.203 |
| | MASE | 1.022 | 1.019 | 1.037 | 1.005 | 0.970 | 0.941 | 1.039 | 0.942 | 4.238 |
| | OWA | 0.946 | 0.944 | 0.956 | 0.936 | 0.905 | 0.895 | 0.957 | 0.891 | 3.067 |
| Others | SMAPE | 7.067 | 7.178 | 6.709 | 6.160 | 4.952 | 4.643 | 6.913 | 5.218 | 10.377 |
| | MASE | 5.587 | 5.302 | 4.953 | 4.713 | 3.347 | 3.128 | 4.507 | 3.892 | 7.864 |
| | OWA | 1.642 | 1.536 | 1.487 | 1.389 | 1.049 | 0.997 | 1.438 | 1.217 | 9.472 |
| Avg. | SMAPE | 12.867 | 12.807 | 13.639 | 12.546 | 12.059 | 11.851 | 12.880 | 11.914 | 5.327 |
| | MASE | 1.887 | 1.864 | 2.095 | 1.740 | 1.623 | 1.587 | 1.836 | 1.603 | 10.285 |
| | OWA | 0.957 | 0.948 | 1.051 | 0.917 | 0.869 | 0.840 | 0.955 | 0.894 | 6.693 |

Table 7. Short-term forecasting results on stock dataset in terms of MAE and MSE, the lower the better. The forecasting horizon is {7, 24}. The better performance in each setting is shown in **bold**.

| Horizon | Metric | TSMixer | + CCM | DLinear | + CCM | PatchTST | + CCM | TimesNet | + CCM | IMP(%) |
|---------|--------|---------|--------------|---------|--------------|--------------|--------------|----------|--------------|--------|
| 7 | MSE | 0.939 | 0.938 | 0.992 | 0.883 | 0.896 | 0.892 | 0.930 | 0.915 | 3.288 |
| | MAE | 0.807 | 0.806 | 0.831 | 0.774 | 0.771 | 0.771 | 0.802 | 0.793 | 2.026 |
| 24 | MSE | 1.007 | 0.991 | 0.996 | 0.917 | 0.930 | 0.880 | 0.998 | 0.937 | 5.252 |
| | MAE | 0.829 | 0.817 | 0.832 | 0.781 | 0.789 | 0.765 | 0.830 | 0.789 | 3.889 |

In table 7, we also observe a significant performance improvement on the stock dataset, achieved by applying CCM. In the stock dataset, we test on new samples (*i.e.*, univariate stock time series) that the model has not seen in the training phase to evaluate the model’s generalization and robustness. By memorizing cluster-specific knowledge from analogous samples, the model potentially captures various market trends and behaviors and thereby makes more accurate and informed forecasting.

5.4. Zero-shot Forecasting Results

Existing time series models tend to be rigidly tailored to specific domains, leading to poor generalization on data from unseen domains. In contrast, CCM leverages learned prototypes to capture cluster-specific knowledge. This enables meaningful comparisons between unseen time series and pre-trained knowledge, facilitating accurate zero-shot forecasting. Following prior work (Jin et al., 2023), we adopt ETT collections (Zhou et al., 2021), where ETTh1 and ETTh2 are hourly recorded, while ETTm1 and ETTm2 are minutely recorded. “1” and “2” indicate two different regions where the datasets originated.

Table 8 shows MSE and MAE results on test datasets. CCM consistently improves the zero-shot forecasting capacity of base time series models in 48 scenarios, including generalization to different domains and different granularities. Specifically, based on the results, we make the following observations. (1) CCM exhibits more significant performance improvement with longer forecasting horizons, highlight-

ing the efficacy of memorizing and leveraging pre-trained knowledge in zero-shot forecasting scenarios. (2) Achieving cross-granularity (*i.e.*, different sampling frequencies) generalization presents greater challenges than cross-domain (*i.e.*, different sampling regions) generalization. However, the result comparison between task ① and ② (or task ④ and ⑤) shows that CCM exhibits superiority in more challenging cross-granularity generalization scenarios. (3) CCM demonstrates a better effect on originally CI base models. For instance, the averaged improvement rates on two CI models, *i.e.*, DLinear and PatchTST, are 10.48% and 11.13% respectively, while the improvement rates on TSMixer and TimesNet are 5.14% and 9.63%. Overall, the experimental results verify the superiority and efficacy of CCM in enhancing zero-shot transfer forecasting.

5.5. Qualitative Visualization

Channel Clustering Visualization. Figure 2 presents the t-SNE visualization of channel and prototype embeddings within ETTh1 and ETTh2 datasets with DLinear as the base model. Each point represents a channel within a sample, with varying colors indicating different channels. In ETTh1 dataset, we discern a pronounced clustering of channels 0, 2, and 4, suggesting that they may be capturing related or redundant information within the dataset. Concurrently, channels 1, 3, 5, and 6 coalesce into another cluster. The similarity matrix in the lower left further corroborates these findings. Clustering is also observable in ETTh2 dataset, particularly among channels 0, 4, and 5, as well as channels 2, 3, and 6. Comparatively, channel 1 shows a dispersion

Table 8. Zero-shot forecasting results on ETT datasets. The forecasting horizon is {96, 720}. The best value in each row is underlined.

| Model Generalization Task | TSMixer + CCM | | | | DLinear + CCM | | | | PatchTST + CCM | | | | TimesNet + CCM | | | | IMP(%) | |
|---------------------------|---------------|-------|-------|--------------|---------------|-------|-------|--------------|----------------|-------|-------|--------------|----------------|-------|-------|--------------|--------------|--------|
| | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | | |
| ① ETTh1→ETTh2 | 96 | 0.288 | 0.357 | <u>0.283</u> | <u>0.353</u> | 0.308 | 0.371 | <u>0.283</u> | <u>0.349</u> | 0.313 | 0.362 | <u>0.292</u> | <u>0.346</u> | 0.391 | 0.412 | <u>0.388</u> | <u>0.410</u> | 3.661 |
| | 720 | 0.374 | 0.414 | <u>0.370</u> | <u>0.413</u> | 0.569 | 0.549 | <u>0.520</u> | <u>0.517</u> | 0.414 | 0.442 | <u>0.386</u> | <u>0.423</u> | 0.540 | 0.508 | <u>0.516</u> | <u>0.491</u> | 4.326 |
| ② ETTh1→ETTm1 | 96 | 0.763 | 0.677 | <u>0.710</u> | <u>0.652</u> | 0.726 | 0.658 | <u>0.681</u> | <u>0.634</u> | 0.729 | 0.667 | <u>0.698</u> | <u>0.647</u> | 0.887 | 0.718 | <u>0.827</u> | <u>0.700</u> | 4.626 |
| | 720 | 1.252 | 0.815 | <u>1.215</u> | <u>0.803</u> | 1.881 | 0.948 | <u>1.138</u> | <u>0.809</u> | 1.459 | 0.845 | <u>1.249</u> | <u>0.795</u> | 1.623 | 0.981 | <u>1.601</u> | <u>0.964</u> | 10.249 |
| ③ ETTh1→ETTm2 | 96 | 0.959 | 0.694 | <u>0.937</u> | <u>0.689</u> | 0.990 | 0.704 | <u>0.896</u> | <u>0.677</u> | 0.918 | 0.694 | <u>0.895</u> | <u>0.677</u> | 1.199 | 0.794 | <u>1.122</u> | <u>0.731</u> | 4.457 |
| | 720 | 1.765 | 0.982 | <u>1.758</u> | <u>0.980</u> | 2.091 | 1.061 | <u>1.681</u> | <u>0.954</u> | 1.925 | 1.014 | <u>1.718</u> | <u>0.966</u> | 2.204 | 1.031 | <u>1.874</u> | <u>1.012</u> | 7.824 |
| ④ ETTh2→ETTh1 | 96 | 0.466 | 0.462 | <u>0.455</u> | <u>0.456</u> | 0.462 | 0.450 | <u>0.427</u> | <u>0.432</u> | 0.620 | 0.563 | <u>0.509</u> | <u>0.495</u> | 0.869 | 0.624 | <u>0.752</u> | <u>0.590</u> | 8.016 |
| | 720 | 0.695 | 0.584 | <u>0.540</u> | <u>0.519</u> | 0.511 | 0.518 | <u>0.484</u> | <u>0.502</u> | 1.010 | 0.968 | <u>0.936</u> | <u>0.686</u> | 1.274 | 0.783 | <u>0.845</u> | <u>0.642</u> | 16.243 |
| ⑤ ETTh2→ETTm2 | 96 | 0.943 | 0.726 | <u>0.876</u> | <u>0.697</u> | 0.736 | 0.656 | <u>0.700</u> | <u>0.642</u> | 0.840 | 0.708 | <u>0.771</u> | <u>0.688</u> | 1.250 | 0.850 | <u>1.064</u> | <u>0.793</u> | 6.344 |
| | 720 | 1.472 | 0.872 | <u>1.464</u> | <u>0.866</u> | 1.813 | 0.938 | <u>1.253</u> | <u>0.844</u> | 1.832 | 1.052 | <u>1.532</u> | <u>0.863</u> | 1.861 | 1.016 | <u>1.671</u> | <u>0.967</u> | 11.439 |
| ⑥ ETTh2→ETTm1 | 96 | 1.254 | 0.771 | <u>1.073</u> | <u>0.714</u> | 1.147 | 0.746 | <u>0.894</u> | <u>0.669</u> | 0.997 | 0.721 | <u>0.789</u> | <u>0.629</u> | 1.049 | 0.791 | <u>0.804</u> | <u>0.657</u> | 16.016 |
| | 720 | 2.275 | 1.137 | <u>1.754</u> | <u>1.065</u> | 1.992 | 1.001 | <u>1.740</u> | <u>0.970</u> | 2.651 | 1.149 | <u>1.695</u> | <u>0.971</u> | 2.183 | 1.103 | <u>1.742</u> | <u>0.983</u> | 15.952 |

among clusters, partly due to its capturing of unique or diverse aspects of the data that do not closely align with the features represented by any clusters. The clustering results demonstrate that CCM not only elucidates the intricate relationships and potential redundancies among the channels but also offers critical insights for feature analysis and enhancing the interpretability of time series models.

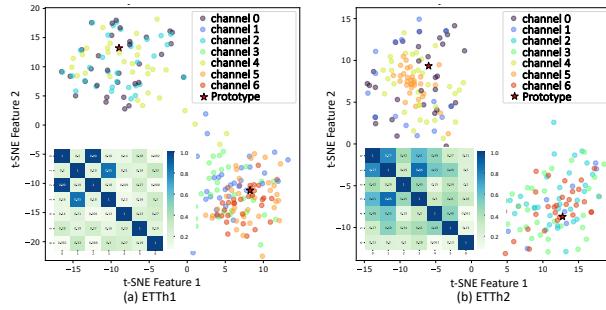


Figure 2. t-SNE visualization of channel and prototype embedding given by DLinear with CCM on (a) ETTh1 and (b) ETTh2 dataset. The lower left corner shows the similarity matrix between channels.

Weight Visualization of Cluster-aware Projection. Figure 3 depicts the weights visualization for the cluster-aware Feed Forward on ETTh1 and ETTm1 datasets, revealing distinct patterns that are indicative of the model’s learned features (Li et al., 2023a; Zeng et al., 2022; Yi et al., 2023). For instance, in the ETTm1 dataset, Cluster 0 shows bright diagonal striping patterns, which may suggest that it is primarily responsible for capturing the most dominant periodic signals in the corresponding cluster. In contrast, Cluster 1 exhibits denser stripes, indicating its role in refining the representation by capturing more subtle or complex periodicities than the first layer does not. The visualization implies the model’s ability to identify and represent periodicity in diverse patterns, which is crucial for time-series forecasting tasks that are characterized by intricate cyclic behaviors.

5.6. Ablation Study on Cluster Number

Figure 4 shows an ablation study on cluster ratios, which is defined as the ratio of the number of clusters to the number of channels. 0.0 means all channels are in a single cluster. We observe that the MSE loss slightly decreases and then increases as the cluster ratio increases, especially for DLinear,

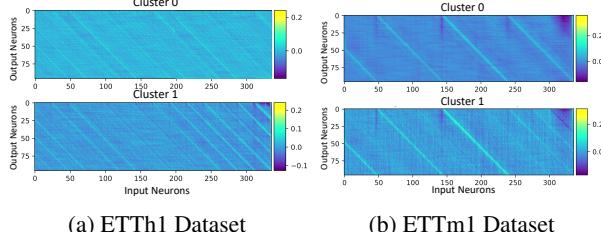


Figure 3. Weights visualization of cluster-wise linear layers on (a) ETTh1 and (b) ETTm1 datasets. The input and output lengths are 336 and 96, respectively. We observe the different periodicities captured by different clusters.

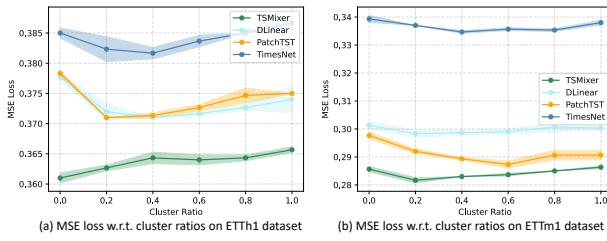


Figure 4. Ablation Study on Cluster Ratios in terms of MSE loss with four base models. The forecasting horizon is 96. (left: ETTh1 dataset; right: ETTm1 dataset)

PatchTST, and TimesNet. Time series models with CCM achieve the best performance when the cluster ratio is in the range of [0.2, 0.6]. It is worth noticing that DLinear and PatchTST, two CI models among four base models, benefit consistently from channel clustering with any number of clusters. More results are provided in Appendix D.

6. Conclusion

This work introduces a novel Channel Clustering Module (CCM) to address the challenge of effective channel management in time series forecasting. CCM strikes a balance between individual channel treatment and capturing cross-channel dependencies by clustering channels based on their intrinsic similarity. Extensive experiments demonstrate the efficacy of CCM in multiple benchmarks. It would be valuable to investigate the applicability of CCM in other domains beyond time series forecasting in future works.

Impact Statement. This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

References

- Ahmed, N. K., Atiya, A. F., Gayar, N. E., and El-Shishiny, H. An empirical comparison of machine learning models for time series forecasting. *Econometric reviews*, 29(5-6): 594–621, 2010.
- Alvarez, F. M., Troncoso, A., Riquelme, J. C., and Ruiz, J. S. A. Energy time series forecasting based on pattern sequence similarity. *IEEE Transactions on Knowledge and Data Engineering*, 23(8):1230–1243, 2010.
- Angryk, R. A., Martens, P. C., Aydin, B., Kempton, D., Mahajan, S. S., Basodi, S., Ahmadzadeh, A., Cai, X., Filali Boubrahimi, S., Hamdi, S. M., et al. Multivariate time series dataset for space weather data analytics. *Scientific data*, 7(1):1–13, 2020.
- Bai, S., Kolter, J. Z., and Koltun, V. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*, 2018.
- Benaouda, D., Murtagh, F., Starck, J.-L., and Renaud, O. Wavelet-based nonlinear multiscale decomposition model for electricity load forecasting. *Neurocomputing*, 70(1-3): 139–154, 2006.
- Chen, L. and Ng, R. On the marriage of l_p -norms and edit distance. In *Proceedings of the Thirtieth international conference on Very large data bases-Volume 30*, pp. 792–803, 2004.
- Chen, S.-A., Li, C.-L., Yoder, N., Arik, S. O., and Pfister, T. Tsmixer: An all-mlp architecture for time series forecasting. *arXiv preprint arXiv:2303.06053*, 2023.
- Cleveland, R. B., Cleveland, W. S., McRae, J. E., and Terpenning, I. Stl: A seasonal-trend decomposition. *J. Off. Stat*, 6(1):3–73, 1990.
- Das, A., Kong, W., Leach, A., Sen, R., and Yu, R. Long-term forecasting with tide: Time-series dense encoder. *arXiv preprint arXiv:2304.08424*, 2023.
- Ding, H., Trajcevski, G., Scheuermann, P., Wang, X., and Keogh, E. Querying and mining of time series data: experimental comparison of representations and distance measures. *Proceedings of the VLDB Endowment*, 1(2): 1542–1552, 2008.
- Feng, A., Chen, J., Garza, J., Berry, B., Salazar, F., Gao, Y., Ying, R., and Tassiulas, L. Efficient high-resolution time series classification via attention kronecker decomposition. *arXiv preprint arXiv:2403.04882*, 2024.
- George, L. and Sumathy, P. An integrated clustering and bert framework for improved topic modeling. *International Journal of Information Technology*, pp. 1–9, 2023.
- Gold, O. and Sharir, M. Dynamic time warping and geometric edit distance: Breaking the quadratic barrier. *ACM Transactions on Algorithms (TALG)*, 14(4):1–17, 2018.
- Gunupudi, R. K., Nimmala, M., Gugulothu, N., and Gali, S. R. Clapp: A self constructing feature clustering approach for anomaly detection. *Future Generation Computer Systems*, 74:417–429, 2017.
- Han, L., Ye, H.-J., and Zhan, D.-C. The capacity and robustness trade-off: Revisiting the channel independent strategy for multivariate time series forecasting. *arXiv preprint arXiv:2304.05206*, 2023.
- Hewamalage, H., Bergmeir, C., and Bandara, K. Recurrent neural networks for time series forecasting: Current status and future directions. *International Journal of Forecasting*, 37(1):388–427, 2021.
- Jang, E., Gu, S., and Poole, B. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.
- Jang, P. A. and Matteson, D. S. Spatial correlation in weather forecast accuracy: a functional time series approach. *Computational Statistics*, pp. 1–15, 2023.
- Ji, J., Wang, J., Huang, C., Wu, J., Xu, B., Wu, Z., Zhang, J., and Zheng, Y. Spatio-temporal self-supervised learning for traffic flow prediction. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pp. 4356–4364, 2023.
- Jiang, J.-Y., Liou, R.-J., and Lee, S.-J. A fuzzy self-constructing feature clustering algorithm for text classification. *IEEE transactions on knowledge and data engineering*, 23(3):335–349, 2010.
- Jin, M., Wang, S., Ma, L., Chu, Z., Zhang, J. Y., Shi, X., Chen, P.-Y., Liang, Y., Li, Y.-F., Pan, S., et al. Time-lm: Time series forecasting by reprogramming large language models. *arXiv preprint arXiv:2310.01728*, 2023.
- Kim, T., Kim, J., Tae, Y., Park, C., Choi, J.-H., and Choo, J. Reversible instance normalization for accurate time-series forecasting against distribution shift. In *International Conference on Learning Representations*, 2021.
- Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

- Kitaev, N., Kaiser, L., and Levskaya, A. Reformer: The efficient transformer. *arXiv preprint arXiv:2001.04451*, 2020.
- Kljun, M. and Tersˇek, M. A review and comparison of time series similarity measures. In *29th International Electrotechnical and Computer Science Conference (ERK 2020). Portoroz*, pp. 21–22, 2020.
- Koprinska, I., Wu, D., and Wang, Z. Convolutional neural networks for energy time series forecasting. In *2018 international joint conference on neural networks (IJCNN)*, pp. 1–8. IEEE, 2018.
- Lai, G., Chang, W.-C., Yang, Y., and Liu, H. Modeling long-and short-term temporal patterns with deep neural networks. *corr abs/1703.07015* (2017). *arXiv preprint arXiv:1703.07015*, 2017.
- Lai, G., Chang, W.-C., Yang, Y., and Liu, H. Modeling long-and short-term temporal patterns with deep neural networks. In *The 41st international ACM SIGIR conference on research & development in information retrieval*, pp. 95–104, 2018.
- Lam, R., Sanchez-Gonzalez, A., Willson, M., Wirnsberger, P., Fortunato, M., Alet, F., Ravuri, S., Ewalds, T., Eaton-Rosen, Z., Hu, W., et al. Learning skillful medium-range global weather forecasting. *Science*, 382(6677):1416–1421, 2023.
- Li, H., Achim, A., and Bull, D. Unsupervised video anomaly detection using feature clustering. *IET signal processing*, 6(5):521–533, 2012.
- Li, L., Su, X., Zhang, Y., Lin, Y., and Li, Z. Trend modeling for traffic time series analysis: An integrated study. *IEEE Transactions on Intelligent Transportation Systems*, 16(6):3430–3439, 2015.
- Li, Z., Qi, S., Li, Y., and Xu, Z. Revisiting long-term time series forecasting: An investigation on linear mapping. *arXiv preprint arXiv:2305.10721*, 2023a.
- Li, Z., Rao, Z., Pan, L., and Xu, Z. Mts-mixers: Multivariate time series forecasting via factorized temporal and channel mixing. *arXiv preprint arXiv:2302.04501*, 2023b.
- Lim, B. and Zohren, S. Time-series forecasting with deep learning: a survey. *Philosophical Transactions of the Royal Society A*, 379(2194):20200209, 2021.
- Liu, G., He, S., Han, X., Luo, Q., Du, R., Fu, X., and Zhao, L. Self-supervised spatiotemporal masking strategy-based models for traffic flow forecasting. *Symmetry*, 15(11):2002, 2023a.
- Liu, M., Zeng, A., Chen, M., Xu, Z., Lai, Q., Ma, L., and Xu, Q. Scinet: Time series modeling and forecasting with sample convolution and interaction. *Advances in Neural Information Processing Systems*, 35:5816–5828, 2022a.
- Liu, S., Yu, H., Liao, C., Li, J., Lin, W., Liu, A. X., and Dustdar, S. Pyraformer: Low-complexity pyramidal attention for long-range time series modeling and forecasting. In *International Conference on Learning Representations*, 2021.
- Liu, Y., Wu, H., Wang, J., and Long, M. Non-stationary transformers: Rethinking the stationarity in time series forecasting. *arXiv preprint arXiv:2205.14415*, 2022b.
- Liu, Y., Hu, T., Zhang, H., Wu, H., Wang, S., Ma, L., and Long, M. itransformer: Inverted transformers are effective for time series forecasting. *arXiv preprint arXiv:2310.06625*, 2023b.
- Liu, Z., Cheng, M., Li, Z., Huang, Z., Liu, Q., Xie, Y., and Chen, E. Adaptive normalization for non-stationary time series forecasting: A temporal slice perspective. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023c.
- Makridakis, S., Spiliotis, E., and Assimakopoulos, V. The m4 competition: Results, findings, conclusion and way forward. *International Journal of Forecasting*, 34(4):802–808, 2018.
- Marin, D., Chang, J.-H. R., Ranjan, A., Prabhu, A., Rastegari, M., and Tuzel, O. Token pooling in vision transformers for image classification. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 12–21, 2023.
- Montero-Manso, P. and Hyndman, R. J. Principles and algorithms for forecasting groups of time series: Locality and globality. *International Journal of Forecasting*, 37(4):1632–1653, 2021.
- Ni, J. and Xu, Y. Forecasting the dynamic correlation of stock indices based on deep learning method. *Computational Economics*, 61(1):35–55, 2023.
- Nie, Y., Nguyen, N. H., Sinthong, P., and Kalagnanam, J. A time series is worth 64 words: Long-term forecasting with transformers. *arXiv preprint arXiv:2211.14730*, 2022.
- Oreshkin, B. N., Carpov, D., Chapados, N., and Bengio, Y. N-beats: Neural basis expansion analysis for interpretable time series forecasting. *arXiv preprint arXiv:1905.10437*, 2019.
- Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al. Pytorch: An imperative style, high-performance

- deep learning library. *Advances in neural information processing systems*, 32, 2019.
- Peiwen, Y. and Changsheng, Z. Is channel independent strategy optimal for time series forecasting? *arXiv preprint arXiv:2310.17658*, 2023.
- Percival, D. B. and Walden, A. T. *Wavelet methods for time series analysis*, volume 4. Cambridge university press, 2000.
- Rangapuram, S. S., Seeger, M. W., Gasthaus, J., Stella, L., Wang, Y., and Januschowski, T. Deep state space models for time series forecasting. *Advances in neural information processing systems*, 31, 2018.
- Salinas, D., Flunkert, V., Gasthaus, J., and Januschowski, T. Deepar: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting*, 36(3):1181–1191, 2020.
- Sen, R., Yu, H.-F., and Dhillon, I. S. Think globally, act locally: A deep neural network approach to high-dimensional time series forecasting. *Advances in neural information processing systems*, 32, 2019.
- Shen, Z., Zhang, M., Zhao, H., Yi, S., and Li, H. Efficient attention: Attention with linear complexities. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pp. 3531–3539, 2021.
- Smyl, S. A hybrid method of exponential smoothing and recurrent neural networks for time series forecasting. *International Journal of Forecasting*, 36(1):75–85, 2020.
- Syarif, I., Prugel-Bennett, A., and Wills, G. Unsupervised clustering approach for network anomaly detection. In *Networked Digital Technologies: 4th International Conference, NDT 2012, Dubai, UAE, April 24-26, 2012. Proceedings, Part I 4*, pp. 135–145. Springer, 2012.
- Tang, B. and Matteson, D. S. Probabilistic transformer for time series analysis. *Advances in Neural Information Processing Systems*, 34:23592–23608, 2021.
- Taylor, S. J. and Letham, B. Forecasting at scale. *The American Statistician*, 72(1):37–45, 2018.
- Team, M. et al. M4 competitor’s guide: prizes and rules. See <https://www.m4.unic.ac.cy/wpcontent/uploads/2018/03/M4-CompetitorsGuide.pdf>, 2018.
- Torres, J. F., Hadjout, D., Sebaa, A., Martínez-Álvarez, F., and Troncoso, A. Deep learning for time series forecasting: a survey. *Big Data*, 9(1):3–21, 2021.
- Tribe, O., Hewamalage, H., Pilyugina, P., Laptev, N., Bergmeir, C., and Rajagopal, R. Neuralprophet: Explainable forecasting at scale, 2021.
- Trindade, A. ElectricityLoadDiagrams20112014. UCI Machine Learning Repository, 2015. DOI: <https://doi.org/10.24432/C58C86>.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.
- Wan, R., Mei, S., Wang, J., Liu, M., and Yang, F. Multivariate temporal convolutional network: A deep neural networks approach for multivariate time series forecasting. *Electronics*, 8(8):876, 2019.
- Wang, H., Peng, J., Huang, F., Wang, J., Chen, J., and Xiao, Y. Micn: Multi-scale local and global context modeling for long-term series forecasting. In *The Eleventh International Conference on Learning Representations*, 2022.
- Wang, S., Li, B. Z., Khabsa, M., Fang, H., and Ma, H. Linformer: Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768*, 2020.
- Wen, Q., Zhang, Z., Li, Y., and Sun, L. Fast robuststl: Efficient and robust seasonal-trend decomposition for time series with complex patterns. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 2203–2213, 2020.
- Wen, Q., Zhou, T., Zhang, C., Chen, W., Ma, Z., Yan, J., and Sun, L. Transformers in time series: A survey. *arXiv preprint arXiv:2202.07125*, 2022.
- Wu, H., Xu, J., Wang, J., and Long, M. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Advances in Neural Information Processing Systems*, 34:22419–22430, 2021.
- Wu, H., Hu, T., Liu, Y., Zhou, H., Wang, J., and Long, M. Timesnet: Temporal 2d-variation modeling for general time series analysis. *arXiv preprint arXiv:2210.02186*, 2022.
- Yi, K., Zhang, Q., Fan, W., Wang, S., Wang, P., He, H., Lian, D., An, N., Cao, L., and Niu, Z. Frequency-domain mlps are more effective learners in time series forecasting. *arXiv preprint arXiv:2311.06184*, 2023.
- Yin, X., Yan, D., Almudaifer, A., Yan, S., and Zhou, Y. Forecasting stock prices using stock correlation graph: A graph convolutional network approach. In *2021 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8. IEEE, 2021.

Yin, Y. and Shang, P. Forecasting traffic time series with multivariate predicting method. *Applied Mathematics and Computation*, 291:266–278, 2016.

Zeng, A., Chen, M., Zhang, L., and Xu, Q. Are transformers effective for time series forecasting? *arXiv preprint arXiv:2205.13504*, 2022.

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

Zhang, H. H., Genton, M. G., and Liu, P. Compactly supported radial basis function kernels. Technical report, North Carolina State University. Dept. of Statistics, 2004.

Zhang, T., Zhang, Y., Cao, W., Bian, J., Yi, X., Zheng, S., and Li, J. Less is more: Fast multivariate time series forecasting with light sampling-oriented mlp structures. *arXiv preprint arXiv:2207.01186*, 2022.

Zhang, Y. and Yan, J. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In *The Eleventh International Conference on Learning Representations*, 2022.

Zhang, Y., Wu, R., Dascalu, S. M., and Harris, F. C. Multi-scale transformer pyramid networks for multivariate time series forecasting. *IEEE Access*, 2024.

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

Zhou, T., Ma, Z., Wen, Q., Sun, L., Yao, T., Yin, W., Jin, R., et al. Film: Frequency improved legendre memory model for long-term time series forecasting. *Advances in Neural Information Processing Systems*, 35:12677–12690, 2022a.

Zhou, T., Ma, Z., Wen, Q., Wang, X., Sun, L., and Jin, R. Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting. *arXiv preprint arXiv:2201.12740*, 2022b.

A. Channel Similarity

Essentially, the similarity between two time series X_i and X_j is defined as $\text{SIM}(X_i, X_j) = \exp\left(\frac{-d(X_i, X_j)}{2\sigma^2}\right)$, where $d(\cdot, \cdot)$ can be any distance metric (Chen & Ng, 2004; Ding et al., 2008), such as Euclidean Distance (L_p), Editing Distance (ED) and Dynamic Time Warping (DTW) (Gold & Sharir, 2018). One may also use other similarity definitions, such as Longest Common Subsequence (LCSS) and Cross-correlation (CCor).

Firstly, the choice of Euclidean distance in this work is motivated by its efficiency and low computational complexity, especially in the case of large datasets or real-time applications. Let H denote the length of the time series. The complexity of the above similarity computation is shown in Table 9.

Secondly, it's worth noting that while there are various similarity computation approaches, studies have demonstrated a strong correlation between Euclidean distance and other distance metrics (Kljun & Ters ek, 2020). This high correlation suggests that, despite different mathematical formulations, these metrics often yield similar results when assessing the similarity between time series. This empirical evidence supports the choice of Euclidean distance as a reasonable approximation of similarity for practical purposes. In our implementation, we select $\sigma = 5$ in Eq. 1 to compute the similarities based on Euclidean distance.

Table 9. Complexity of similarity computation

| Euclidean | Edit Distance | DTW | LCSS | CCor |
|------------------|--------------------|--------------------|--------------------|--------------------|
| $\mathcal{O}(H)$ | $\mathcal{O}(H^2)$ | $\mathcal{O}(H^2)$ | $\mathcal{O}(H^2)$ | $\mathcal{O}(H^2)$ |

B. Multivariate and Univariate Adaptation

We provide pseudocodes for training time series models enhanced with CCM in Algorithm 1. Algorithm 2 displays pseudocodes for the inference phase, where both the training and test sets have the same number of channels. The components in the pretrained model \mathcal{F} , highlighted in blue, remain fixed during the inference phase. It's important to note that zero-shot forecasting in Algorithm 2 is adaptable to various scenarios. Let's discuss these scenarios:

- **Training on a univariate dataset and inferring on either univariate or multivariate samples:** In this case, the model learns prototypes from a vast collection of univariate time series in the training set. As a result, the model can effortlessly adapt to forecasting unseen univariate time series in a zero-shot manner. To forecast unseen multivariate time series, we decompose each multivariate sample into multiple univariate samples, where each univariate sample can be processed by the pretrained model. The future multivariate time series can be obtained by stacking multiple future univariate time series.
- **Training on a multivariate dataset and inferring on either univariate or multivariate samples:** For Channel-Dependent models, test samples should have the same number of channels as the training samples, as seen in sub-datasets within ETT collections (Zhou et al., 2021). In contrast, for Channel-Independent models, zero-shot forecasting can be performed on either univariate or multivariate samples, even when they have different numbers of channels.

C. Experiments

C.1. Datasets

Public Datasets. We utilize nine commonly used datasets for long-term forecasting evaluation. Firstly, ETT collection (Zhou et al., 2021), which documents the oil temperature and load features of electricity transformers over the period spanning July 2016 to July 2018. This dataset is further subdivided into four sub-datasets, ETTHs and ETTms, with hourly and 15-minute sampling frequencies, respectively. s can be 1 or 2, indicating two different regions. Electricity dataset (Trindade, 2015) encompasses electricity consumption data from 321 clients from July 2016 to July 2019. Exchange (Lai et al., 2017) compiles daily exchange rate information from 1990 to 2016. Traffic dataset contains hourly traffic load data from 862 sensors in San Francisco areas from 2015 to 2016. Weather dataset offers a valuable resource with 21 distinct weather indicators, including air temperature and humidity, collected every 10 minutes throughout the year 2021. ILI documents the weekly ratio of influenza-like illness patients relative to the total number of patients, spanning from 2002 to 2021. Dataset statistics can be found in Table 10.

We adopt M4 dataset for short-term forecasting evaluation, which involves 100,000 univariate time series collected from different domains, including finance, industry, etc. The M4 dataset is further divided into 6 sub-datasets, according to the

Algorithm 1 Forward function of time series models with channel clustering module. C is the number of channels in the dataset. K is the number of clusters. T is the length of historical data. H is the forecasting horizon.

Input: Historical time series $X \in \mathbb{R}^{T \times C}$
Output: Future time series $Y \in \mathbb{R}^{H \times C}$; Prototype embedding $\mathbf{C} \in \mathbb{R}^{K \times d}$

Initialize the weights of K linear layer θ_k for $k = 1, \dots, K$
 Initialize K cluster embedding $c_k \in \mathbb{R}^d$ for $k = 1, \dots, K$. ▷ Cluster Embedding \mathbf{C}

$X \leftarrow \text{Normalize}(X)$
 $\mathbf{S}_{i,j} \leftarrow \exp\left(\frac{-\|X_i - X_j\|^2}{2\sigma^2}\right)$. ▷ Compute Similarity Matrix \mathbf{S}
 $h_i \leftarrow \text{MLP}(X_i)$ for each channel i . ▷ Channel Embedding \mathbf{H} via MLP Encoder in the Cluster Assigner

$p_{i,k} \leftarrow \text{Normalize}\left(\frac{c_k^\top h_i}{\|c_k\| \|h_i\|}\right) \in [0, 1]$. ▷ Compute Clustering Probability Matrix \mathbf{P}
 $\mathbf{M} \leftarrow \text{Bernoulli}(\mathbf{P})$. ▷ Sample Clustering Membership Matrix \mathbf{M}

$\mathbf{C} \leftarrow \text{Normalize}\left(\exp\left(\frac{(W_Q \mathbf{C})(W_K \mathbf{H})^\top}{\sqrt{d}}\right) \odot \mathbf{M}^\top\right) W_V \mathbf{H}$. ▷ Update Cluster Embedding \mathbf{C} via Cross Attention

$\hat{\mathbf{H}} = \text{Temporal Module}(\mathbf{H})$. ▷ Update via Temporal Modules

for channel i in $\{1, 2, \dots, C\}$ **do**
 $Y_i \leftarrow h_{\theta^i}(\hat{\mathbf{H}}_i)$ where $\theta^i = \sum_k p_{i,k} \theta_k$. ▷ Weight Averaging and Projection

end for

Algorithm 2 Zero-shot forecasting of time series models with channel clustering module. C is the number of channels in both the training and test datasets. K is the number of clusters. T is the length of historical data. H is the forecasting horizon.

Input: Historical time series $X \in \mathbb{R}^{T \times C}$; Pretrained Models \mathcal{F}
Output: Future time series $Y \in \mathbb{R}^{H \times C}$;

Load the weights of K linear layer θ_k for $k = 1, \dots, K$ from \mathcal{F}
 Load K cluster embedding $c_k \in \mathbb{R}^d$ for $k = 1, \dots, K$ from \mathcal{F} . ▷ Cluster Embedding \mathbf{C}

$X \leftarrow \text{Normalize}(X)$
 $\mathbf{S}_{i,j} \leftarrow \exp\left(\frac{-\|X_i - X_j\|^2}{2\sigma^2}\right)$. ▷ Compute Similarity Matrix \mathbf{S}
 $h_i \leftarrow \text{MLP}(X_i)$ for each channel i . ▷ Channel Embedding \mathbf{H} via MLP Encoder in the Cluster Assigner

$p_{i,k} \leftarrow \text{Normalize}\left(\frac{c_k^\top h_i}{\|c_k\| \|h_i\|}\right) \in [0, 1]$. ▷ Compute Clustering Probability Matrix \mathbf{P}
 $\mathbf{M} \leftarrow \text{Bernoulli}(\mathbf{P})$. ▷ Sample Clustering Membership Matrix \mathbf{M}

$\hat{\mathbf{H}} = \text{Temporal Module}(\mathbf{H})$. ▷ Update via Temporal Modules

for channel i in $\{1, 2, \dots, C\}$ **do**
 $Y_i \leftarrow h_{\theta^i}(\hat{\mathbf{H}}_i)$ where $\theta^i = \sum_k p_{i,k} \theta_k$. ▷ Weight Averaging and Projection

end for

sampling frequency.

Stock Dataset. We design a new time series benchmark dataset constructed from publicly available stock-market data. We deploy commercial stock market API to probe the market statistics for 1390 stocks spanning 10 years from Nov.25, 2013 to Sep.1, 2023. We collect stock price data from 9:30 a.m. to 4:00 p.m. every stock open day except early closure days. The sampling granularity is set to be 5 minutes. Missing record is recovered by interpolation from nearby timestamps and all stock time series are processed to have aligned timestep sequences. We implement short-term forecasting on the stock dataset, which is more focused on market sentiment, and short-term events that can cause stock prices to fluctuate over days, weeks, or months. Thereby, we set the forecasting horizon as 7 and 24.

C.2. Metrics

Following standard evaluation protocols (Wu et al., 2022), we utilize the Mean Absolute Error (MAE) and Mean Square Error(MSE) for long-term and stock price forecasting. The Symmetric Mean Absolute Percentage Error (SMAPE), Mean Absolute Scaled Error (MASE), and Overall Weighted Average (OWA) are used as metrics for M4 dataset (Makridakis et al., 2018; Oreshkin et al., 2019). The formulations are shown in Eq. 5. Let y_t and \hat{y}_t denote the ground-truth and the forecast at the t -th timestep, respectively. H is the forecasting horizon. In M4 dataset, MASE is a standard scale-free metric, where s is

Table 10. The statistics of dataset in long-term and short-term forecasting tasks

| Tasks | Dataset | Channels | Forecast Horizon | Length | Frequency | Domain |
|------------|--------------|----------|---------------------|--------|-----------|----------------|
| Long-term | ETTh1 | 7 | {96, 192, 336, 720} | 17420 | 1 hour | Temperature |
| | ETTh2 | 7 | {96, 192, 336, 720} | 17420 | 1 hour | Temperature |
| | ETTm1 | 7 | {96, 192, 336, 720} | 69680 | 15 min | Temperature |
| | ETTm2 | 7 | {96, 192, 336, 720} | 69680 | 15 min | Temperature |
| | Illness | 7 | {96, 192, 336, 720} | 966 | 1 week | Illness Ratio |
| | Exchange | 8 | {96, 192, 336, 720} | 7588 | 1 day | Exchange Rates |
| | Weather | 21 | {96, 192, 336, 720} | 52696 | 10 min | Weather |
| | Electricity | 321 | {96, 192, 336, 720} | 26304 | 1 hour | Electricity |
| | Traffic | 862 | {96, 192, 336, 720} | 17544 | 1 hour | Traffic Load |
| Short-term | M4-Yearly | 1 | 6 | 23000 | yearly | Demographic |
| | M4 Quarterly | 1 | 8 | 24000 | quarterly | Finance |
| | M4 Monthly | 1 | 18 | 48000 | monthly | Industry |
| | M4 Weekly | 1 | 13 | 359 | weekly | Macro |
| | M4 Daily | 1 | 14 | 4227 | daily | Micro |
| | M4 Hourly | 1 | 48 | 414 | hourly | Other |
| | Stock | 1 | {7, 24} | 10000 | 5 min | Stock |

the periodicity of the data (*e.g.*, 12 for monthly recorded sub-dataset) (Makridakis et al., 2018). MASE measures the scaled error *w.r.t.* the naïve predictor that simply copies the historical records of s periods in the past. Instead, SMAPE scales the error by the average between the forecast and ground truth. Particularly, OWA is an M4-specific metric (Team et al., 2018) that assigns different weights to each metric.

$$\begin{aligned}
 \text{MAE} &= \frac{1}{H} \sum_{t=1}^H |\mathbf{y}_t - \hat{\mathbf{y}}_t|, & \text{MSE} &= \frac{1}{H} \sum_{i=1}^H (\mathbf{y}_t - \hat{\mathbf{y}}_t)^2, \\
 \text{SMAPE} &= \frac{200}{H} \sum_{i=1}^H \frac{|\mathbf{y}_t - \hat{\mathbf{y}}_t|}{|\mathbf{y}_t| + |\hat{\mathbf{y}}_t|}, & \text{MASE} &= \frac{1}{H} \sum_{i=1}^H \frac{|\mathbf{y}_t - \hat{\mathbf{y}}_t|}{\frac{1}{H-s} \sum_{j=s+1}^H |\mathbf{y}_j - \mathbf{y}_{j-s}|}, \\
 \text{OWA} &= \frac{1}{2} \left[\frac{\text{SMAPE}}{\text{SMAPE}_{\text{Naive2}}} + \frac{\text{MASE}}{\text{MASE}_{\text{Naive2}}} \right]
 \end{aligned} \tag{5}$$

C.3. Experiment Details

To verify the superiority of CCM in enhancing the performance of mainstream time series models, we select four popular and state-of-the-art models, including linear models such as TSMixer (Chen et al., 2023), DLinear (Zeng et al., 2022), transformer-based model PatchTST (Nie et al., 2022) and convolution-based model TimesNet (Wu et al., 2022). We build time series models using their official codes and optimal model configuration¹²³⁴.

In the data preprocessing stage, we apply reversible instance normalization (Kim et al., 2021) to ensure zero mean and unit standard deviation, avoiding the time series distribution shift. Forecasting loss is MSE for long-term forecasting datasets and the stock dataset. Instead, we use SMAPE loss for M4 dataset. We select Adam (Kingma & Ba, 2014) with the default hyperparameter configuration for (β_1, β_2) as $(0.9, 0.999)$. An early-stopping strategy is used to mitigate overfitting. The experiments are conducted on a single NVIDIA RTX A6000 48GB GPU, with PyTorch (Paszke et al., 2019) framework. The code is available at the following anonymous link: <https://anonymous.4open.science/r/TimeSeriesCCM-ECC2>. We use the official codes and follow the best model configuration to implement the base models. Then we apply CCM to the base models, keeping the hyperparameters unchanged for model backbones. Experiment configurations, including the number of MLP layers in the cluster assigner, the layer number in the temporal module, hidden dimension, the best cluster number, and regularization parameter β on nine real-world datasets are shown in Table 11.

¹<https://github.com/yuqinie98/PatchTST>

²<https://github.com/cure-lab/LTSF-Linear>

³<https://github.com/google-research/google-research/tree/master/tsmixer>

⁴<https://github.com/thuml/TimesNet>

Table 11. Experiment configuration.

| | # clusters | β | # linear layers in MLP | hidden dimension | # layers (TSMixer) | # layers (PatchTST) | # layers (TimesNet) |
|-------------|------------|---------|------------------------|------------------|--------------------|---------------------|---------------------|
| ETTh1 | 2 | 0.3 | 1 | 128 | 2 | 2 | 3 |
| ETTm1 | 2 | 0.3 | 1 | 64 | 2 | 4 | 2 |
| ETTh2 | 2 | 0.3 | 1 | 64 | 2 | 4 | 3 |
| ETTm2 | 2 | 0.9 | 1 | 24 | 2 | 4 | 4 |
| Exchange | 2 | 0.9 | 1 | 32 | 2 | 4 | 3 |
| ILY | 2 | 0.9 | 1 | 36 | 2 | 6 | 3 |
| Weather | [2,5] | 0.5 | 2 | 64 | 4 | 3 | 3 |
| Electricity | [3,10] | 0.5 | 2 | 128 | 4 | 3 | 3 |
| Traffic | [3,10] | 0.5 | 2 | 128 | 4 | 3 | 3 |

C.4. Comparison between CCM and Other Approach

 Table 12. Full Results on Comparison between CCM and existing regularization method for enhanced performance on CI/CD strategies in terms of MSE metric. The best results are highlighted in **bold**.

| | | CD | CI | +PRReg | +CCM |
|-----------------|-------------|--------------|-------|--------------|--------------|
| ETTh1(48) | Linear | 0.402 | 0.345 | 0.342 | 0.342 |
| | Transformer | 0.861 | 0.655 | 0.539 | 0.518 |
| ETTh2(48) | Linear | 0.711 | 0.226 | 0.239 | 0.237 |
| | Transformer | 1.031 | 0.274 | 0.273 | 0.284 |
| ETTm1(48) | Linear | 0.404 | 0.354 | 0.311 | 0.310 |
| | Transformer | 0.458 | 0.379 | 0.349 | 0.300 |
| ETTm2(48) | Linear | 0.161 | 0.147 | 0.136 | 0.146 |
| | Transformer | 0.281 | 0.148 | 0.144 | 0.143 |
| Exchange(48) | Linear | 0.119 | 0.051 | 0.042 | 0.042 |
| | Transformer | 0.511 | 0.101 | 0.044 | 0.048 |
| Weather(48) | Linear | 0.142 | 0.169 | 0.131 | 0.130 |
| | Transformer | 0.251 | 0.168 | 0.180 | 0.164 |
| ILY(24) | Linear | 2.343 | 2.847 | 2.299 | 2.279 |
| | Transformer | 5.309 | 4.307 | 3.254 | 3.206 |
| Electricity(48) | Linear | 0.195 | 0.196 | 0.196 | 0.195 |
| | Transformer | 0.250 | 0.185 | 0.185 | 0.183 |

Predict Residuals with Regularization. Prior work (Han et al., 2023) demonstrates that the main drawback of CD models is their inclination to generate sharp and non-robust forecasts, deviating from the actual trend. Thereby, Predict Residuals with Regularization (PRReg for simplicity), a specifically designed regularization objective, is proposed to improve the robustness of CD methods as follows.

$$\mathcal{L} = \frac{1}{N} \sum_{j=1}^N \mathcal{L}_F \left(f \left(X^{(j)} - A^{(j)} \right) + A^{(j)}, Y^{(j)} \right) + \lambda \Omega(f), \quad (6)$$

where the superscript j indicates the sample index. \mathcal{L}_F is MSE loss. $A^{(j)} = X_{L,:}^j$ represents the last values of each channel in $X^{(j)}$. Therefore, the objective encourages the model to generate predictions that are close to the nearby historical data and keep the forecasts smooth and robust. The regularization term Ω (L_2 norm in practice) further restricts the complexity of the model and ensures smoothness in the predictions. It was demonstrated that PRReg can achieve better performance than original CD and CI strategies (Han et al., 2023). We conduct extensive experiments on long-term forecasting benchmarks to compare PRReg and CCM. The full results are shown in Table 12. The numbers in parentheses next to the method represent the forecasting horizon. We set the length of the look-back window to 36 for ILY and 96 for other datasets for consistency with previous works (Han et al., 2023). The base models are linear model (Zeng et al., 2022) and basic transformer (Vaswani et al., 2017). The values in the PRReg column are the best results with any λ , collected from (Han et al., 2023). We observe from Table 12 that CCM successfully improves forecasting performance on original CI/CD strategies (or reached comparable results) in 13 out of 16 settings, compared with PRReg method.

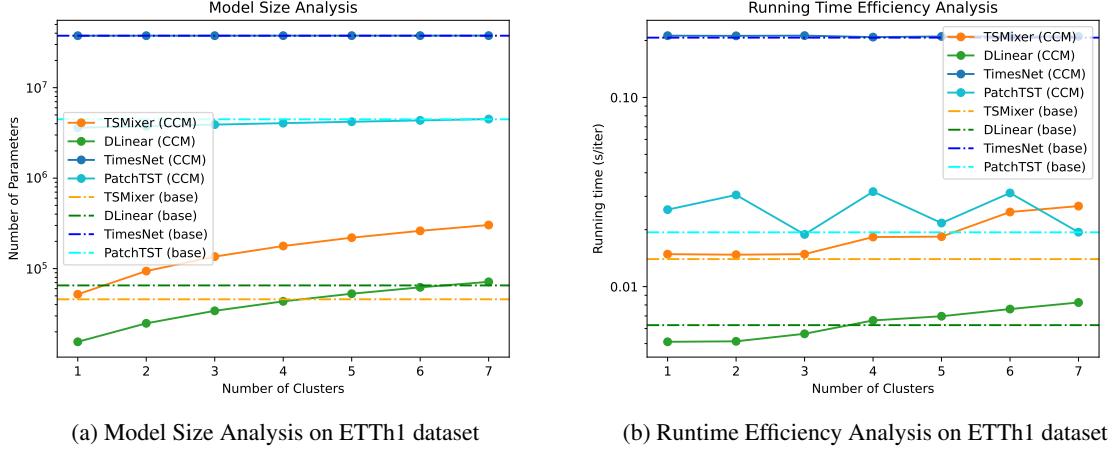


Figure 5. Efficiency analysis in model size and running time on ETTh1 dataset.

Table 13. Standard deviation of Table 2 on long-term forecasting benchmarks. The forecasting horizon is 96.

| | | ETTh1 | ETTm1 | ETTh2 | ETTm2 | Exchange | IL1 | Weather | Electricity | Traffic |
|----------|-----|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|
| TSMixer | MSE | 0.361±0.001 | 0.285±0.001 | 0.284±0.001 | 0.171±0.001 | 0.089±0.004 | 1.914±0.031 | 0.149±0.008 | 0.142±0.002 | 0.376±0.006 |
| | MAE | 0.392±0.001 | 0.339±0.001 | 0.343±0.002 | 0.260±0.001 | 0.209±0.009 | 0.879±0.009 | 0.198±0.009 | 0.237±0.004 | 0.264±0.005 |
| +CCM | MSE | 0.365±0.001 | 0.283±0.002 | 0.278±0.001 | 0.167±0.001 | 0.085±0.006 | 1.938±0.015 | 0.147±0.007 | 0.139±0.005 | 0.375±0.006 |
| | MAE | 0.393±0.001 | 0.337±0.002 | 0.338±0.002 | 0.260±0.001 | 0.206±0.011 | 0.874±0.012 | 0.194±0.009 | 0.235±0.008 | 0.262±0.005 |
| DLinear | MSE | 0.375±0.002 | 0.299±0.001 | 0.289±0.001 | 0.167±0.001 | 0.088±0.006 | 2.215±0.031 | 0.192±0.011 | 0.153±0.004 | 0.411±0.006 |
| | MAE | 0.399±0.001 | 0.343±0.001 | 0.353±0.001 | 0.260±0.001 | 0.215±0.010 | 1.081±0.009 | 0.250±0.008 | 0.239±0.005 | 0.284±0.005 |
| +CCM | MSE | 0.371±0.001 | 0.298±0.001 | 0.285±0.001 | 0.166±0.002 | 0.085±0.006 | 1.935±0.034 | 0.187±0.015 | 0.142±0.003 | 0.411±0.005 |
| | MAE | 0.393±0.001 | 0.343±0.002 | 0.348±0.02 | 0.258±0.002 | 0.214±0.013 | 0.935±0.012 | 0.245±0.020 | 0.247±0.006 | 0.282±0.004 |
| PatchTST | MSE | 0.375±0.003 | 0.294±0.003 | 0.278±0.003 | 0.174±0.003 | 0.094±0.008 | 1.593±0.016 | 0.149±0.008 | 0.138±0.004 | 0.360±0.005 |
| | MAE | 0.398±0.004 | 0.351±0.004 | 0.340±0.004 | 0.261±0.003 | 0.216±0.012 | 0.757±0.015 | 0.198±0.012 | 0.233±0.005 | 0.249±0.005 |
| +CCM | MSE | 0.371±0.002 | 0.289±0.005 | 0.274±0.006 | 0.168±0.003 | 0.088±0.006 | 1.561±0.021 | 0.147±0.008 | 0.136±0.002 | 0.357±0.007 |
| | MAE | 0.396±0.003 | 0.338±0.005 | 0.336±0.006 | 0.256±0.003 | 0.208±0.009 | 0.750±0.009 | 0.197±0.013 | 0.231±0.006 | 0.246±0.006 |
| TimesNet | MSE | 0.384±0.005 | 0.338±0.006 | 0.340±0.005 | 0.187±0.005 | 0.107±0.009 | 2.317±0.024 | 0.172±0.011 | 0.168±0.002 | 0.593±0.010 |
| | MAE | 0.402±0.005 | 0.375±0.006 | 0.374±0.005 | 0.267±0.003 | 0.234±0.013 | 0.934±0.010 | 0.220±0.013 | 0.272±0.005 | 0.321±0.008 |
| +CCM | MSE | 0.380±0.004 | 0.335±0.005 | 0.336±0.003 | 0.189±0.003 | 0.105±0.006 | 2.139±0.038 | 0.169±0.015 | 0.158±0.003 | 0.554±0.009 |
| | MAE | 0.400±0.004 | 0.371±0.006 | 0.371±0.005 | 0.270±0.005 | 0.231±0.010 | 0.936±0.018 | 0.215±0.024 | 0.259±0.006 | 0.316±0.008 |

C.5. Complexity and Efficiency Analysis

We evaluate the model size and runtime efficiency of the proposed CCM with various numbers of clusters on ETTh1 dataset, as shown in Figure 5. The batch size is 32, and the hidden dimension is 64. We keep all other hyperparameters consistent to ensure fair evaluation. It is worth noting that CCM reduces the model complexity based on Channel-Independent models (*e.g.*, PatchTST, DLinear), since CCM essentially uses cluster identity to replace channel identity. The generalizability of CI models is thereby enhanced as well. When it comes to Channel-Dependent models, CCM increases the model complexity with negligible overhead, considering the improved forecasting performance.

C.6. Error Bar

Experimental results in this paper are averaged from five runs with different random seeds. We report the standard deviation for base models and CCM-enhanced versions on long-term forecasting datasets in Table 13, M4 dataset in Table 14 and stock dataset in Table 15.

Table 14. Standard deviation of Table 6 on M4 dataset

| Model | | TSMixer | + CCM | DLinear | + CCM | PatchTST | + CCM | TimesNet | + CCM |
|----------|-------|---------|-------|---------|-------|----------|-------|----------|-------|
| Yearly | SMAPE | 0.122 | 0.130 | 0.087 | 0.089 | 0.135 | 0.134 | 0.168 | 0.162 |
| | MASE | 0.022 | 0.022 | 0.019 | 0.017 | 0.018 | 0.021 | 0.0017 | 0.017 |
| | OWA | 0.002 | 0.002 | 0.002 | 0.002 | 0.007 | 0.009 | 0.010 | 0.011 |
| Quaterly | SMAPE | 0.101 | 0.103 | 0.100 | 0.100 | 0.079 | 0.073 | 0.106 | 0.105 |
| | MASE | 0.016 | 0.016 | 0.015 | 0.015 | 0.008 | 0.009 | 0.013 | 0.011 |
| | OWA | 0.008 | 0.007 | 0.006 | 0.008 | 0.013 | 0.016 | 0.009 | 0.009 |
| Monthly | SMAPE | 0.113 | 0.113 | 0.110 | 0.111 | 0.122 | 0.120 | 0.120 | 0.134 |
| | MASE | 0.013 | 0.015 | 0.009 | 0.013 | 0.017 | 0.019 | 0.011 | 0.012 |
| | OWA | 0.001 | 0.001 | 0.002 | 0.001 | 0.003 | 0.002 | 0.004 | 0.004 |
| Others | SMAPE | 0.113 | 0.110 | 0.126 | 0.128 | 0.137 | 0.130 | 0.129 | 0.125 |
| | MASE | 0.024 | 0.026 | 0.036 | 0.031 | 0.023 | 0.025 | 0.028 | 0.025 |
| | OWA | 0.011 | 0.013 | 0.009 | 0.008 | 0.023 | 0.019 | 0.024 | 0.020 |
| Avg. | SMAPE | 0.113 | 0.115 | 0.111 | 0.103 | 0.136 | 0.134 | 0.148 | 0.153 |
| | MASE | 0.027 | 0.025 | 0.021 | 0.017 | 0.026 | 0.021 | 0.027 | 0.042 |
| | OWA | 0.006 | 0.004 | 0.004 | 0.004 | 0.013 | 0.016 | 0.021 | 0.036 |

Table 15. Standard deviation of Table 7 on stock dataset

| Horizon | Metric | TSMixer | + CCM | DLinear | + CCM | PatchTST | + CCM | TimesNet | + CCM |
|---------|--------|---------|-------|---------|-------|----------|-------|----------|-------|
| 7 | MSE | 0.001 | 0.001 | 0.001 | 0.001 | 0.003 | 0.003 | 0.004 | 0.004 |
| | MAE | 0.001 | 0.001 | 0.001 | 0.001 | 0.003 | 0.002 | 0.003 | 0.003 |
| 24 | MSE | 0.002 | 0.002 | 0.001 | 0.002 | 0.005 | 0.004 | 0.005 | 0.005 |
| | MAE | 0.001 | 0.001 | 0.001 | 0.001 | 0.003 | 0.002 | 0.003 | 0.004 |

D. Ablation Study

D.1. Influence of Cluster Ratio

The number of clusters is an important hyperparameter in the CCM method. To verify the effectiveness of our design, we conduct an ablation study for all base models on four long-term forecasting datasets. The full results are shown in Table 16. We tune different cluster ratios, defined as the ratio of the number of clusters to the number of channels. *Original* means the original base model without any channel clustering mechanism. *0.0* indicates grouping all channels into the same cluster. We make the following observations. (1) For most cases, the channel clustering module (CCM) with any number of clusters greater than 1 consistently improves the forecasting performance upon base models. (2) For Channel-Independent base models, such as DLinear and PatchTST, grouping all channels into one cluster results in a channel fusion at the last layer, leading to a degradation in forecasting performance than the original CI models. (3) For most cases, the cluster ratio in the range of [0.2, 0.6] typically results in the best performance.

D.2. Influence of Look-back Window Length

In this section, we conduct additional ablation experiments to investigate the effect of different look-back window lengths in the newly collected stock dataset, which determines how much historical information the time series model uses to make short-term forecasts. Specifically, the ablation study helps identify the risk of overfitting or underfitting based on the chosen look-back window length. An overly long window may lead to overfitting, while a short window may cause underfitting. Table 17 display the forecasting performance on forecasting horizon 7 and 24. The length of the look-back window ranges from two to four times the forecasting horizon. From Table 17, we make the following observations. (1) CCM effectively improves the base models' forecasting performance in 24 cases across different base models, forecasting horizons, and look-back window lengths consistently. (2) Especially, CCM achieves better enhancement when the look-back window is shorter.

Table 16. Sensitivity of cluster ratio in terms of MSE metric. The forecasting horizon is 96. 0.0 means grouping all channels into the same cluster. *original* means the base model without the CCM mechanism.

| | Cluster Ratio | Original | 0.0 | 0.2 | 0.4 | 0.6 | 0.8 | 1.0 |
|-------------|---------------|----------|-------|-------|-------|-------|-------|-------|
| ETTh1 | TSMixer | 0.361 | 0.361 | 0.362 | 0.365 | 0.363 | 0.364 | 0.366 |
| | DLinear | 0.375 | 0.378 | 0.371 | 0.371 | 0.372 | 0.372 | 0.371 |
| | PatchTST | 0.375 | 0.380 | 0.372 | 0.371 | 0.373 | 0.376 | 0.375 |
| | TimesNet | 0.384 | 0.384 | 0.380 | 0.383 | 0.385 | 0.385 | 0.388 |
| ETTm1 | TSMixer | 0.285 | 0.285 | 0.283 | 0.283 | 0.284 | 0.285 | 0.286 |
| | DLinear | 0.299 | 0.303 | 0.298 | 0.298 | 0.299 | 0.300 | 0.300 |
| | PatchTST | 0.294 | 0.298 | 0.292 | 0.289 | 0.289 | 0.293 | 0.293 |
| | TimesNet | 0.338 | 0.338 | 0.337 | 0.335 | 0.336 | 0.335 | 0.337 |
| Exchange | TSMixer | 0.089 | 0.089 | 0.086 | 0.087 | 0.088 | 0.090 | 0.092 |
| | DLinear | 0.088 | 0.093 | 0.088 | 0.087 | 0.085 | 0.089 | 0.089 |
| | PatchTST | 0.094 | 0.095 | 0.089 | 0.088 | 0.088 | 0.091 | 0.093 |
| | TimesNet | 0.107 | 0.107 | 0.105 | 0.105 | 0.107 | 0.107 | 0.107 |
| Electricity | TSMixer | 0.142 | 0.142 | 0.139 | 0.139 | 0.140 | 0.143 | 0.143 |
| | DLinear | 0.153 | 0.160 | 0.143 | 0.142 | 0.143 | 0.147 | 0.150 |
| | PatchTST | 0.138 | 0.142 | 0.136 | 0.136 | 0.138 | 0.140 | 0.140 |
| | TimesNet | 0.168 | 0.168 | 0.160 | 0.159 | 0.167 | 0.168 | 0.169 |

Table 17. Short-term forecasting on stock dataset with different look-back window length in {14, 21, 28}. The forecasting length is 7. The best results with the same base model are underlined. **Bold** means CCM successfully enhances forecasting performance over the base model.

| Forecast | 7 | | | | | | 24 | | | | | |
|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|
| | 14 | | 21 | | 28 | | 48 | | 72 | | 96 | |
| Input Length | MSE | MAE |
| TSMixer | 0.947 | 0.806 | 0.974 | 0.816 | 0.939 | 0.807 | 1.007 | 0.829 | 1.016 | 0.834 | 1.100 | 0.856 |
| + CCM | 0.896 | 0.778 | 0.954 | 0.808 | 0.938 | 0.806 | 0.991 | 0.817 | 1.006 | 0.824 | 1.078 | 0.851 |
| DLinear | 1.003 | 0.834 | 0.995 | 0.833 | 0.992 | 0.831 | 0.998 | 0.832 | 0.996 | 0.832 | 0.998 | 0.832 |
| + CCM | 0.897 | 0.778 | 0.904 | 0.782 | 0.883 | 0.774 | 0.921 | 0.786 | 0.917 | 0.781 | 0.969 | 0.798 |
| PatchTST | 0.933 | 0.804 | 0.896 | 0.771 | 0.926 | 0.794 | 0.976 | 0.793 | 0.951 | 0.790 | 0.930 | 0.789 |
| + CCM | 0.931 | 0.758 | 0.892 | 0.771 | 0.924 | 0.790 | 0.873 | 0.767 | 0.860 | 0.759 | 0.880 | 0.765 |
| TimesNet | 0.943 | 0.816 | 0.934 | 0.803 | 0.930 | 0.802 | 0.998 | 0.830 | 1.003 | 0.818 | 1.013 | 0.821 |
| + CCM | 0.926 | 0.796 | 0.911 | 0.789 | 0.915 | 0.793 | 0.937 | 0.789 | 0.974 | 0.803 | 0.979 | 0.804 |
| Imp. (%) | 4.492 | 4.590 | 3.527 | 2.211 | 3.230 | 2.152 | 6.492 | 3.798 | 5.344 | 3.271 | 3.409 | 2.445 |

E. Visualization Results

E.1. Channel-wise Performance and Channel Similarity

Figure 6 illustrates the channel-wise forecasting performance in terms of MSE metric and channel similarity on ETTh1 dataset with DLinear. We use the model's performance difference on the original dataset and the randomly shuffled dataset as the model's fitting ability on a specific channel. Note that MSE loss is computed on channels that have been standardized, which means that any scaling differences between them have been accounted for. Figure 6 highlights a noteworthy pattern: when two channels exhibit a higher degree of similarity, there tends to be a corresponding similarity in the performance on these channels. This observation suggests that channels with similar characteristics tend to benefit similarly from the optimization. It implies a certain level of consistency in the improvement of MSE loss when dealing with similar channels.

F. Discussion

This paper presents the Channel Clustering Module (CCM) for enhanced performance of time series forecasting models, aiming to balance the treatment of individual channels while capturing essential cross-channel interactions. Despite its promising contributions, there still exist limitations and directions for future work that warrant consideration.

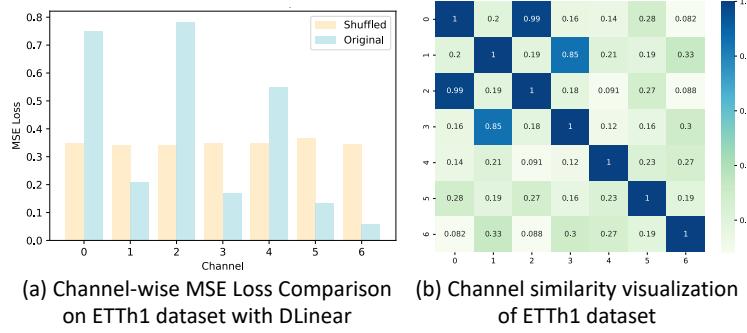


Figure 6. (a) Channel-wise forecasting performance and (b) Channel similarity on ETTh1 dataset illustrate the correlation between model performance and intrinsic similarity

Limitation. While CCM shows improvements in forecasting, its scalability to extremely large datasets remains to be tested. Moreover, the clustering and embedding processes in CCM introduce additional computational overhead. The efficiency of CCM in real-time forecasting scenarios, where computational resources are limited, requires further optimization.

Future Works. Future research can focus on adapting CCM to specific domains, such as biomedical signal processing or geospatial data analysis, where external domain-specific knowledge can be involved in the similarity computation. Furthermore, exploring alternative approaches to develop a dynamical clustering mechanism within CCM could potentially enhance forecasting efficacy. It is also worth examining the effectiveness of CCM in contexts beyond time series forecasting.