# Sequence Complementor: Complementing Transformers for Time Series Forecasting with Learnable Sequences

**Xiwen Chen[1*], Peijie Qiu[2*], Wenhui Zhu[3*], Huayu Li[4], Hao Wang[1], Aristeidis Sotiras[2], Yalin Wang[3], Abolfazl Razi[1]**

[1] Clemson University,
[2] Washington University in St. Louis,
[3]Arizona State University,
[4] The University of Arizona
{xiwenc, hao9}@g.clemson.edu, {peijie.qiu, aristeidis.sotiras}@wustl.edu,
{wzhu59, ylwang}@asu.edu, hl459@arizona.edu, arazi@clemson.edu

## Abstract

Since its introduction, the transformer has shifted the development trajectory away from traditional models (e.g., RNN, MLP) in time series forecasting, which is attributed to its ability to capture global dependencies within temporal tokens. Follow-up studies have largely involved altering the tokenization and self-attention modules to better adapt Transformers for addressing special challenges like non-stationarity, channel-wise dependency, and variable correlation in time series. However, we found that the expressive capability of sequence representation is a key factor influencing Transformer performance in time forecasting after investigating several representative methods, where there is an almost linear relationship between sequence representation entropy and mean square error, with more diverse representations performing better. In this paper, we propose a novel attention mechanism with Sequence Complementors and prove feasible from an information theory perspective, where these learnable sequences are able to provide complementary information beyond current input to feed attention. We further enhance the Sequence Complementors via a diversification loss that is theoretically covered. The empirical evaluation of both long-term and short-term forecasting has confirmed its superiority over the recent state-of-the-art methods.

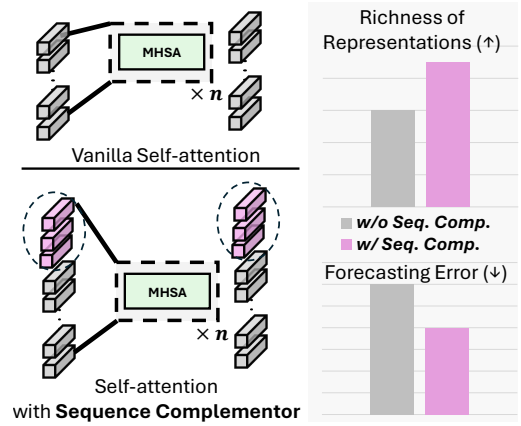



Figure 1: The vanilla self-attention mechanism v.s. the self-attention mechanism with proposed learnable `Sequence Complementors`, which serve as complementary sequences to the original input sequence (**Left**). The integration of Sequence Complementors results in richer learned representations and a better forecasting performance (**Right**).

## Introduction

Time series forecasting (TSF) plays a crucial role in various domains, enabling the extraction of meaningful patterns and the prediction of future events based on historical data. It incubates a wide spectrum of applications, ranging from financial risk assessment, weather forecasting, and energy sustainability to healthcare (Wang et al. 2024b). Recently, the introduction of transformers (Vaswani et al. 2017) has dramatically shifted the trajectory of model designs for TSF largely due to their ability to capture long-range dependencies.

Following this trend, numerous efforts have been devoted to enhancing the architectural designs of transformer-based TSF models (Zhou et al. 2021; Liu et al. 2021; Zhou et al. 2022; Wu et al. 2021; Liu et al. 2022b, 2024). Although they have achieved satisfactory performance, their effectiveness in TSF is empirically challenged even by simple linear models (Zeng et al. 2023; Lin et al. 2024). However, an in-depth analysis of what limits the performance of transformers for TSF is insufficient in the literature. This motivates us to investigate the fundamental limits of transformers for TSF in both an empirical and theoretical way.

For this purpose, we first conduct a series of empirical observations on four different types of transformers for TSF. We observe that the learned latent representations in four different types of transformers are not rich and generalizable enough, measured by the rank and entropy of the learned feature embeddings. Interestingly, we find that this has a direct relationship with the performance of the TSF task: more diverse and richer latent representations typically lead to higher performance. We believe that this phenomenon is likely due to the combination of limited training samples and domain shifts in most TSF datasets.

*These authors contributed equally.

Based on these empirical observations, the most direct solution to enhance the performance of transformers for TSF is to learn rich and generalizable representations. However, this typically requires a larger scale dataset containing diverse samples (Bengio, Courville, and Vincent 2013; LeCun, Bengio, and Hinton 2015), which is impractical for our tasks. This inspires us to consider artificially enriching the dataset by injecting a learnable sequence into the raw input sequence. We term this mechanism as `Sequence Complementor`, as it complements transformers to discover additional information from the input sequence by interacting with the complementary sequence in a self-attention (see Fig. 1). Although this solution is simple, it has nice information-theoretic guarantees in lowering the bound of mean-squared error and explicitly increasing the diversity of the learned latent representations. To further enforce the learnable complementary sequence to be diverse, we propose a diversification loss by leveraging the volume maximization of sub-matrices. Our extensive empirical evaluations across a variety of TSF tasks demonstrate the effectiveness of the proposed method.

In summary, our contributions are threefold: **(i)** We demonstrate a strong correlation between forecasting performance and the learned representations of different transformer variants; **(ii)** To this end, we propose a theoretically sound `Sequence Complementor` to enrich the learned representations of Transformer; **(iii)** We further propose a novel loss to diversify the learnable Sequence Complementor from a theoretical perspective.

## Preliminaries

Without loss of generality, the multivariate time series forecasting (MTSF) given historical observations $\boldsymbol{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{T_{in}}\} \in \mathbb{R}^{T_{in} \times N}$ with $T$ time steps and $N$ variates, the goal is to predict the future $T_{out}$ time steps $\boldsymbol{Y} = \{\boldsymbol{x}_{T_{in}+1}, \ldots, \boldsymbol{x}_{T_{in}+T_{out}}\} \in \mathbb{R}^{T_{out} \times N}$. We consider transformer-based models with a general structure for MTSF (Wu et al. 2021; Liu et al. 2022b; Nie et al. 2023):

$$\boldsymbol{X} \xrightarrow{f_{\texttt{p}}} \boldsymbol{Z}_0 \xrightarrow{f_{\texttt{embed}}} \boldsymbol{Z}_{embed} \xrightarrow{f_{\texttt{Enc}}} \boldsymbol{Z}_{enc} \xrightarrow{f_{\texttt{Dec}}} \hat{\boldsymbol{Y}}, \quad (1)$$

where we denote the feature of each sequence $\boldsymbol{X}$ after patchifying as $\boldsymbol{Z}_0 \in \mathbb{R}^{L \times D_0}$, with $L$ and $D_0$ being the number of patchified sequences and the initial dimension of patches, respectively. $\boldsymbol{Z}_{embed} \in \mathbb{R}^{L \times D}$ denotes the embedded patches. Likewise, $\boldsymbol{Z}_{enc} \in \mathbb{R}^{L \times D}$ denotes the latent feature learned by the encoder, and $\hat{\boldsymbol{Y}}$ denotes the final prediction. The encoder often involves several transformer blocks. It is worth mentioning that early works (e.g., (Wu et al. 2021)) prefer to utilize transformer blocks as the encoder, while recent works (Nie et al. 2023) abandon transformer-based decoder and prefer to directly apply a linear projector. In this sense, we also consider the linear projector as `encoder` without the loss of generalizability.

## The Analysis of Transformers in Time Series Forecasting

The transformer usually suffers information redundancy, which leads to a large number of neurons highly similar (Dalvi et al. 2020; Bhojanapalli et al. 2021; Dai et al. 2020). Some works on network pruning studies typically address the issue of avoiding computational resource waste by considering the removal of neurons through channel shrinkage (Zhao et al. 2019; Wen et al. 2016; Liu et al. 2017). Here, we conjecture that there is a relationship between this information redundancy and the performance of transformer-based methods in MTSF. To validate our hypothesis, we first select several representative models, including the vanilla transformer, Autoformer (Wu et al. 2021), Non-stationary Transformer (Liu et al. 2022b), and PatchTST (Nie et al. 2023). Under fair experimental conditions (Appendix C), we visualized feature similarity by selecting the best-performing variant of each model. Specifically, for each model, we calculated the patch-wise similarity after the self-attention block, generating a feature similarity map for each model (Fig. 2(**Left**)). An intriguing observation is that PatchTST, which exhibits relatively lower feature correlation, consistently achieves better performance (lower MSE scores). This suggests that models with less information redundancy, or lower feature similarity and higher diversity, tend to perform better. However, drawing conclusions based solely on visual inspection of feature correlation is insufficient. Here, we introduce two metrics that are used to further analyze the relation between feature diversity and performance ($\boldsymbol{Z}_{enc}$): (i) the ratio of the dominant singular values in $\boldsymbol{Z}_{enc}$ and (ii) the entropy of features. Given the singular values of $\boldsymbol{Z}_{enc}$ (i.e., $\{\sigma_1, \sigma_2, \ldots, \cdots\}$), the ratio of the dominant singular value is calculated as the ration between the dominant singular values ($\sigma_i > 0.1$) and the total number of singular values. Below, we discuss how to compute entropy:

**Definition 1.** Let $\boldsymbol{Z} \in \mathbb{R}^d$ be a random vector that follows a multivariate Gaussian distribution with covariance matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{d \times d}$. The Gaussian entropy of $\boldsymbol{Z}$ is defined as:

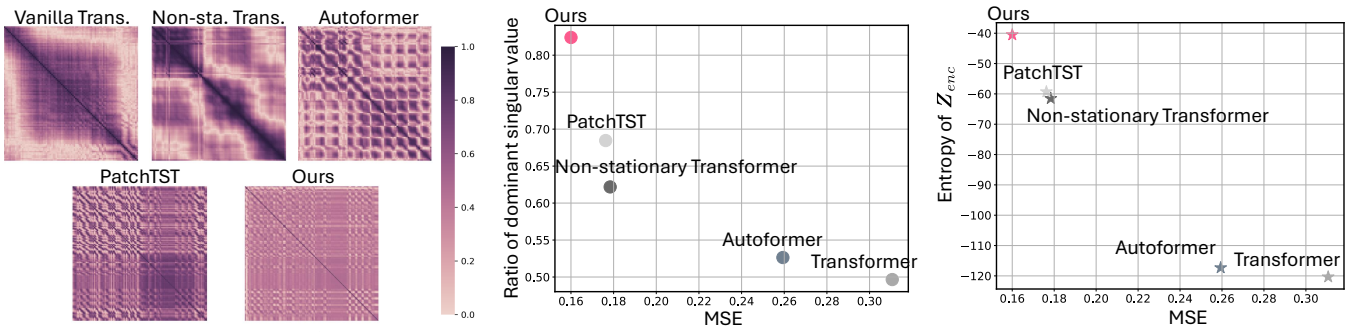
$$H_{\mathcal{G}}(\boldsymbol{Z}) = \frac{1}{2} \log \left( (2\pi e)^d \det(\boldsymbol{\Sigma}) \right). \quad (2)$$

*Remark* 1. The entropy can be estimated by a set of samples $\boldsymbol{Z} = [\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots, \boldsymbol{z}_n]^\top \in \mathbb{R}^{n \times d}$, where each $\boldsymbol{z}_i$ is an $n$-dimensional vector, the Gaussian entropy is computed as:

$$H_{\mathcal{G}}(\boldsymbol{Z}) = \frac{1}{2} \log \left( (2\pi e)^d \det \left( \frac{1}{n} \boldsymbol{Z}^\top \boldsymbol{Z} + \epsilon \boldsymbol{I} \right) \right), \quad (3)$$

where the covariance $\boldsymbol{\Sigma}$ in Eq. 2 is approximated by $\frac{1}{n} \sum_{i=1}^{n} \boldsymbol{z}_i \boldsymbol{z}_i^T = \frac{1}{n} \boldsymbol{Z}^\top \boldsymbol{Z}$. $\epsilon$ and $\boldsymbol{I}$ denote a very small number and identity matrix, respectively, which are used to avoid non-trivial solutions. This approximation is adopted by (Ma et al. 2007; Yu et al. 2020; Chen et al. 2024, 2025).

In this analysis, we employ the Gaussian entropy as a surrogate estimation of the richness or diversity of $\boldsymbol{Z}_{enc}$, and a higher entropy corresponds to a richer or more diverse feature representation. We consider patches to come from one sequence as a set of samples. The metrics are measured when the network is well-trained. Our analysis result reveals insights that a negative linear relationship between feature entropy and the ratio of the dominant singular value with MSE (refer to Fig. 2(**Center**,**Right**)). Specifically, higher feature entropy and a greater ratio of the dominant singular value generally correspond to better performance, as indicated by




(a) Feature correlation (b) Ratio of dominant singular value (c) Entropy of $\boldsymbol{Z}_{enc}$

Figure 2: The analysis of transformers for time series forecasting: (a) the correlation of the learned latent representations from the encoder ($\boldsymbol{Z}_{enc}$), (b) the ratio of dominant singular value against MSE, and (c) the feature entropy against MSE.

lower MSE. In summary, the performance of transformers in TSF is largely constrained by the lack of richness in the learned latent representations, as they struggle to generalize effectively to the domain shifts that are commonly encountered in TSF. In contrast, diverse feature representations typically contribute to better generalization (Bengio, Courville, and Vincent 2013; LeCun, Bengio, and Hinton 2015), which often leads to better performance in TSF tasks.

## Sequence Complementor

Our empirical analysis suggests that the key to enhancing the transformer's performance in time series forecasting revolves around learning rich feature representations. In this section, we answer this question with the proposed sequence complementor, which involves adding learnable complementary sequences to a raw time-sequence input. Although the proposed method is simple, it is surprisingly effective and has nice theoretical guarantees.

### Learnable Complementary Sequence

The most effective way to learn rich feature representations is to train the model on more samples (Bengio, Courville, and Vincent 2013; LeCun, Bengio, and Hinton 2015; Sun et al. 2017). However, this is not always feasible for TSF in practice, where we typically have a variety of TSF tasks, each of which contains only limited samples. An intuitive idea for this challenge is to improve the diversity of the raw input time sequence. To this end, we propose a learnable complementary sequence that adds an additional learnable sequence to the raw input sequence to enhance its diversity.

The implementation of the proposed learnable complementary sequence is straightforward. Following the definition of the transformer in Eq. 1, we concatenate $K$ number of learnable sequences $\boldsymbol{S}$ to the feature embeddings $\boldsymbol{Z}_0$:

$$\tilde{\boldsymbol{Z}}_0 \in \mathbb{R}^{(L_c+K)\times P} \leftarrow \texttt{concat}(\boldsymbol{Z}_0, \boldsymbol{S}), \tag{4}$$

where $\texttt{concat}(\cdot,\cdot)$ denotes concatenation operation, and $L_c$ and $P$ denotes the number of patches for each channel and patchified sequence length, respectively. Since $\boldsymbol{S}$ is learnable, adding $\boldsymbol{S}$ directly to $\boldsymbol{Z}_0$ is more straightforward than adding it to the raw input sequence $\boldsymbol{X}$. Although this process can be model-agnostic and seamlessly integrated into most current TSF transformers, the detailed implementation can vary from model to model (e.g., different patchifying strategies will result in different numbers of tokens). As a proof of concept, we consider the patchifying strategy outlined in (Nie et al. 2023) as an example in this paper. This will result in $L_c = (T_{in} - P)/S + 2$ number of patches, where $S$ denotes the stride used for patchifying.

We then add one more embedding layer to project $\tilde{\boldsymbol{Z}}_0$ to an embedding feature $\boldsymbol{Z}_{embed}$, which is commonly used in many transformer design (Vaswani et al. 2017):

$$\boldsymbol{Z}_{embed} \in \mathbb{R}^{(L_c+K)\times D} \leftarrow f_{\texttt{linear}}(\tilde{\boldsymbol{Z}}_0), \tag{5}$$

where $D$ denotes the embedding dimensions. It is noteworthy that the positional encoding $f_{\texttt{pos}}$ is only applied to patches from $\boldsymbol{X}$ to preserve the relations of original positions:

$$\begin{aligned}\boldsymbol{Z}_{embed}[:L_c] &\leftarrow f_{\texttt{pos}}(\boldsymbol{Z}_{embed}[:L_c]) + \boldsymbol{Z}_{embed}[:L_c],\\ \boldsymbol{Z}_{embed} &\leftarrow \texttt{concat}(\boldsymbol{Z}_{embed}[:L_c], \boldsymbol{Z}_{embed}[L_c+1:]).\end{aligned} \tag{6}$$

Then, we can feed $\boldsymbol{Z}_{embed}$ to the transformer blocks of the encoder $f_{\texttt{enc}}$ to obtain latent representations $\boldsymbol{Z}_{enc}$:

$$\boldsymbol{Z}_{enc} \in \mathbb{R}^{(L_c+K)\times D} \leftarrow f_{\texttt{enc}}(\boldsymbol{Z}_{embed}). \tag{7}$$

We want to reiterate that we do not intend to modify the original network architecture and also to reduce the computation overhead, we only use the patches from the original sequence for the subsequent operations. Since there is enough interaction among complementary patches and original patches through a self-attention mechanism, there would not be substantial information loss if we abandon complementary patches here. For simplicity, we here only show the self-attention mechanism on the last transformer block from the encoder to reveal this fact:

$$\begin{aligned}\boldsymbol{Z}_{enc}[:L_c] &= \text{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{d_k}}\right)\boldsymbol{V},\\ \boldsymbol{Q} &= \boldsymbol{Z}_{embed}[:L_c]\boldsymbol{W}_Q,\\ \boldsymbol{K} &= [\boldsymbol{Z}_{embed}[:L_c], \boldsymbol{Z}_{embed}[L_c+1:]]\boldsymbol{W}_K,\\ \boldsymbol{V} &= [\boldsymbol{Z}_{embed}[:L_c], \boldsymbol{Z}_{embed}[L_c+1:]]\boldsymbol{W}_V.\end{aligned} \tag{8}$$

where $\boldsymbol{W}_Q, \boldsymbol{W}_k, \boldsymbol{W}_V$ are learnable parameters. Then the subsequent operations is presented as,

$$\hat{\boldsymbol{Y}} = \texttt{reshape}(f_{\texttt{dec}}(\boldsymbol{Z}_{enc}[:L_c]). \tag{9}$$

**Complexity analysis.** Adding complementary sequences will not introduce a huge computation overhead in self-attention. Specifically, this process increases the computational complexity from $\mathcal{O}((L_c^2)$ to $\mathcal{O}((L_c+K)(L_c+K)) \approx \mathcal{O}(L_c^2+K^2)$. Additionally, the number of newly introduced parameters is negligible- only $K \times P$ for each channel, where $k$ and $P$ are often small (e.g. $K=3, P=16$). Hence, the overall newly introduced computational and memory complexity is acceptable.

## Theoretical Justification

Below, we justify this joint from an information-theoretic perspective.

**Justification 1.** We first note that including the sequence complementors has the potential to result in a richer representation. This is because it can increase the entropy $H(\boldsymbol{Z}_{enc})$:

$$H(\boldsymbol{Z}_{enc}) \leq H(\boldsymbol{Z}_{enc}, \boldsymbol{S}). \tag{10}$$

*Proof.* Please refer to Appendix A. □

As suggested by previous observations, a higher entropy $H(\boldsymbol{Z}_{enc})$ indicates diverse feature representations, which shows a strong relationship to forecasting performance.

**Justification 2.** We then note that a richer representation $H(\boldsymbol{Z}_{enc})$ can potentially lead to a lower forecasting error. Here, we first show the relations between MSE and the conditional uncertainty $H(\boldsymbol{Y}|\boldsymbol{Z}_{enc})$,

*Lemma* 1. Under Gaussian assumption, the minimum mean-squared error (MMSE), is bounded by,

$$\text{MMSE} \geq \frac{\exp 2H(\boldsymbol{Y}|\boldsymbol{Z}_{enc})}{2\pi e}. \tag{11}$$

Here, $H(\cdot|\cdot)$ denotes the conditional entropy.

*Proof.* Please refer to Appendix A. □

*Remark* 2. Lemma 1 is supported by our earlier observations that diverse and rich latent representations ($\boldsymbol{Z}_{enc}$) generally result in a lower MSE. This reduction in MSE stems from a decrease in conditional entropy $H(\boldsymbol{Y}|\boldsymbol{Z}_{enc})$, which reduces the uncertainty of $\boldsymbol{Y}$ given $\boldsymbol{Z}_{enc}$.

Therefore, by adding the complementary sequence, we may proactively lead to a lower conditional entropy,

*Theorem* 1. The integration of the proposed complementary sequence lowers the bound of MMSE:

$$H(\boldsymbol{Y}|\boldsymbol{Z}_{enc}, \boldsymbol{S}) \leq H(\boldsymbol{Y}|\boldsymbol{Z}_{enc}). \tag{12}$$

*Proof.* Please refer to Appendix A. □

These two justifications converge to our intuitive observations, where the entropy $H(\boldsymbol{Z}_{enc})$ denotes the amount of knowledge we know and conditional entropy $H(\boldsymbol{Y}|\boldsymbol{Z}_{enc})$ presents the uncertainty about $\boldsymbol{Y}$ after knowing $\boldsymbol{Z}_{enc}$. Adding more information to the system, in this case, through complementary sequences, enriches our knowledge about $\boldsymbol{Y}$, and hence potentially leads to improved performance.

## Diversified Complementary Sequence

We can further diversify or orthogonalize the learnable complementary sequence to enforce them to inject more information. Although it is easy to initialize the Sequence Complementors from a set of orthogonal bases, they may not necessarily be orthogonal during network optimization. As a consequence, they are likely to converge to a trivial solution where all complementary sequences are highly similar or identical. To tackle this issue, we propose a diversification loss to diversify the complementary sequences by leveraging the volume maximization of sub-matrices (Kulesza, Taskar et al. 2012; Petit, Roumy, and Maugey 2023). Different from their problems, which try to find an optimized set of indices that maximizes the volume (i.e., a discrete optimization), we propose a differentiable loss that can be learned through back-propagation with any automatic differentiation framework.

**Definition 2.** For given Sequence Complementors $\boldsymbol{S} \in \mathbb{R}^{K \times P}$, the volume is defined as,

$$\text{vol}(\boldsymbol{S}) = \prod_{i=1}^{K} (\sigma_{\boldsymbol{S}})_i, \tag{13}$$

where $(\sigma_{\boldsymbol{S}})_i$ is the $i$-th largest non-zero singular value of $\boldsymbol{S}$.

*Theorem* 2. If $\forall \|\boldsymbol{S}_i\| = 1, i \in [K]$ holds, maximizing $\text{vol}(\boldsymbol{S})$ results in $\boldsymbol{S}_i \perp \boldsymbol{S}_j$, for all $\forall i \neq j, i, j \in [K]$.

*Proof.* This can be proven via the arithmetic mean-geometric mean (AM-GM) inequality. Please refer to Appendix A. □

**Learning objective.** To avoid the product being an arbitrarily large number that burdens the computation, we take the logarithm to $\text{vol}(\boldsymbol{S})$, which leads to the proposed diversification loss $\mathcal{L}_{dcs}$ for learning complementary sequence:

$$\mathcal{L}_{dcs}(\boldsymbol{S}) = -\sum_{i}^{K} 2\log((\sigma_{\boldsymbol{S}})_i + \epsilon), \quad \text{s.t. } \forall \|\boldsymbol{S}_i\| = 1,$$

where $\epsilon$ is a very small number to avoid the logarithmic operation being negative infinite, and apparently a smaller $\mathcal{L}_{dcs}(\boldsymbol{S})$ denotes a larger volume. Computing this loss only requires $\mathcal{O}(K^2P)$ due to the SVD decomposition, which is negligible. We compute the diversification loss independently for each channel in parallel to minimize the computational cost. The overall objective function is a weighted combination of MSE loss and the proposed diversification loss:

$$\mathcal{L}_{obj} = \mathcal{L}_{mse}(\hat{\boldsymbol{Y}}, \boldsymbol{Y}) + \lambda_{dcs}\mathcal{L}_{dcs}(\boldsymbol{S}), \tag{14}$$

where $\lambda_{dcs} > 0$ is the balance weight. This is a tunable hyper-parameter and we set it to 0.1 for all experiments. The overall algorithmic summary is presented in Appendix C.

# Experiments and Results

**Experimental setup.** We evaluate our proposed architecture on both **long-term forecasting** task and **short-term forecasting** task with 8 datasets and 6 datasets, respectively. It is worth mentioning that each dataset in the long-term setting

| Model | | Ours | | iTransformer | | Rlinear | | PatchTST | | Crossformer | | TimesNet | | Dlinear | | SCINet | | FEDformer | | Stationary | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Metric | | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE | MSE | MAE |
| ETTm1 | 96 | **0.321** | **0.359** | 0.334 | 0.368 | 0.355 | 0.376 | 0.329 | 0.367 | 0.404 | 0.426 | 0.338 | 0.375 | 0.345 | 0.372 | 0.418 | 0.438 | 0.379 | 0.419 | 0.386 | 0.398 |
| | 192 | **0.362** | 0.386 | 0.377 | 0.391 | 0.391 | 0.392 | 0.367 | **0.385** | 0.450 | 0.451 | 0.374 | 0.387 | 0.380 | 0.389 | 0.439 | 0.450 | 0.426 | 0.441 | 0.459 | 0.444 |
| | 336 | **0.393** | **0.406** | 0.426 | 0.420 | 0.424 | 0.415 | 0.399 | 0.410 | 0.532 | 0.515 | 0.410 | 0.411 | 0.413 | 0.413 | 0.490 | 0.485 | 0.445 | 0.459 | 0.495 | 0.464 |
| | 720 | **0.450** | 0.442 | 0.491 | 0.459 | 0.487 | 0.450 | 0.454 | **0.439** | 0.666 | 0.589 | 0.478 | 0.450 | 0.474 | 0.453 | 0.595 | 0.550 | 0.543 | 0.490 | 0.585 | 0.516 |
| | Avg. | **0.381** | **0.398** | 0.407 | 0.410 | 0.414 | 0.408 | 0.387 | 0.400 | 0.513 | 0.496 | 0.400 | 0.406 | 0.403 | 0.407 | 0.485 | 0.481 | 0.448 | 0.452 | 0.481 | 0.456 |
| ETTm2 | 96 | **0.172** | **0.256** | 0.180 | 0.264 | 0.182 | 0.265 | 0.175 | 0.259 | 0.287 | 0.366 | 0.187 | 0.267 | 0.193 | 0.292 | 0.286 | 0.377 | 0.203 | 0.287 | 0.192 | 0.274 |
| | 192 | **0.236** | **0.299** | 0.250 | 0.309 | 0.246 | 0.304 | 0.241 | 0.302 | 0.414 | 0.492 | 0.249 | 0.309 | 0.284 | 0.362 | 0.399 | 0.445 | 0.269 | 0.328 | 0.280 | 0.339 |
| | 336 | **0.298** | **0.339** | 0.311 | 0.348 | 0.307 | 0.342 | 0.305 | 0.343 | 0.597 | 0.542 | 0.321 | 0.351 | 0.369 | 0.427 | 0.637 | 0.591 | 0.325 | 0.366 | 0.334 | 0.361 |
| | 720 | **0.395** | **0.397** | 0.412 | 0.407 | 0.407 | 0.398 | 0.402 | 0.400 | 1.730 | 1.042 | 0.408 | 0.403 | 0.554 | 0.522 | 0.960 | 0.735 | 0.421 | 0.415 | 0.417 | 0.413 |
| | Avg. | **0.275** | **0.323** | 0.288 | 0.332 | 0.286 | 0.327 | 0.281 | 0.326 | 0.757 | 0.610 | 0.291 | 0.333 | 0.350 | 0.401 | 0.571 | 0.537 | 0.305 | 0.349 | 0.306 | 0.347 |
| ETTh1 | 96 | **0.375** | **0.394** | 0.386 | 0.405 | 0.386 | 0.395 | 0.414 | 0.419 | 0.423 | 0.448 | 0.384 | 0.402 | 0.386 | 0.400 | 0.654 | 0.599 | 0.376 | 0.419 | 0.513 | 0.491 |
| | 192 | 0.423 | 0.425 | 0.441 | 0.436 | 0.437 | **0.424** | 0.460 | 0.445 | 0.471 | 0.474 | 0.436 | 0.429 | 0.437 | 0.432 | 0.719 | 0.631 | **0.420** | 0.448 | 0.534 | 0.504 |
| | 336 | **0.457** | 0.448 | 0.487 | 0.458 | 0.479 | **0.446** | 0.501 | 0.466 | 0.570 | 0.546 | 0.491 | 0.469 | 0.481 | 0.459 | 0.778 | 0.659 | 0.459 | 0.465 | 0.588 | 0.535 |
| | 720 | **0.462** | 0.472 | 0.503 | 0.491 | 0.481 | **0.470** | 0.500 | 0.488 | 0.653 | 0.621 | 0.521 | 0.500 | 0.519 | 0.516 | 0.836 | 0.699 | 0.506 | 0.507 | 0.643 | 0.616 |
| | Avg. | **0.429** | 0.435 | 0.454 | 0.448 | 0.446 | **0.434** | 0.469 | 0.454 | 0.529 | 0.522 | 0.458 | 0.450 | 0.456 | 0.452 | 0.747 | 0.647 | 0.440 | 0.460 | 0.570 | 0.537 |
| ETTh2 | 96 | **0.283** | **0.337** | 0.297 | 0.349 | 0.288 | 0.338 | 0.302 | 0.348 | 0.745 | 0.584 | 0.340 | 0.374 | 0.333 | 0.387 | 0.707 | 0.621 | 0.358 | 0.397 | 0.476 | 0.458 |
| | 192 | **0.362** | **0.388** | 0.380 | 0.400 | 0.374 | 0.390 | 0.388 | 0.400 | 0.877 | 0.656 | 0.402 | 0.414 | 0.477 | 0.476 | 0.860 | 0.689 | 0.429 | 0.439 | 0.512 | 0.493 |
| | 336 | **0.408** | **0.425** | 0.428 | 0.432 | 0.415 | 0.426 | 0.426 | 0.433 | 1.043 | 0.731 | 0.452 | 0.452 | 0.594 | 0.541 | 1.000 | 0.744 | 0.496 | 0.487 | 0.552 | 0.551 |
| | 720 | **0.419** | 0.441 | 0.427 | 0.445 | 0.420 | **0.440** | 0.431 | 0.446 | 1.104 | 0.763 | 0.462 | 0.468 | 0.831 | 0.657 | 1.249 | 0.838 | 0.463 | 0.474 | 0.562 | 0.560 |
| | Avg. | **0.368** | **0.398** | 0.383 | 0.407 | 0.374 | 0.399 | 0.387 | 0.407 | 0.942 | 0.684 | 0.414 | 0.427 | 0.559 | 0.515 | 0.954 | 0.723 | 0.437 | 0.449 | 0.526 | 0.516 |
| ECL | 96 | 0.156 | 0.252 | **0.148** | **0.240** | 0.201 | 0.281 | 0.181 | 0.270 | 0.219 | 0.314 | 0.168 | 0.272 | 0.197 | 0.282 | 0.247 | 0.345 | 0.193 | 0.308 | 0.169 | 0.273 |
| | 192 | 0.175 | 0.269 | **0.162** | **0.253** | 0.201 | 0.283 | 0.188 | 0.274 | 0.231 | 0.322 | 0.184 | 0.289 | 0.196 | 0.285 | 0.257 | 0.355 | 0.201 | 0.315 | 0.182 | 0.286 |
| | 336 | 0.190 | 0.284 | **0.178** | **0.269** | 0.215 | 0.298 | 0.204 | 0.293 | 0.246 | 0.337 | 0.198 | 0.300 | 0.209 | 0.301 | 0.269 | 0.369 | 0.214 | 0.329 | 0.200 | 0.304 |
| | 720 | 0.246 | 0.324 | 0.225 | **0.317** | 0.257 | 0.331 | 0.246 | 0.324 | 0.280 | 0.363 | **0.220** | 0.320 | 0.245 | 0.333 | 0.299 | 0.390 | 0.246 | 0.355 | 0.222 | 0.321 |
| | Avg. | 0.192 | 0.282 | **0.178** | **0.270** | 0.219 | 0.298 | 0.205 | 0.290 | 0.244 | 0.334 | 0.192 | 0.295 | 0.212 | 0.300 | 0.268 | 0.365 | 0.214 | 0.327 | 0.193 | 0.296 |
| Exchange | 96 | **0.082** | **0.200** | 0.086 | 0.206 | 0.093 | 0.217 | 0.088 | 0.205 | 0.256 | 0.367 | 0.107 | 0.234 | 0.088 | 0.218 | 0.267 | 0.396 | 0.148 | 0.278 | 0.111 | 0.237 |
| | 192 | **0.175** | **0.297** | 0.177 | 0.299 | 0.184 | 0.307 | 0.176 | 0.299 | 0.470 | 0.509 | 0.226 | 0.344 | 0.176 | 0.315 | 0.351 | 0.459 | 0.271 | 0.315 | 0.219 | 0.335 |
| | 336 | 0.325 | 0.413 | 0.331 | 0.419 | 0.351 | 0.432 | **0.301** | **0.397** | 1.268 | 0.883 | 0.367 | 0.448 | 0.313 | 0.427 | 1.324 | 0.853 | 0.460 | 0.427 | 0.421 | 0.476 |
| | 720 | 0.840 | **0.690** | 0.847 | 0.691 | 0.886 | 0.714 | 0.901 | 0.714 | 1.767 | 1.068 | 0.964 | 0.746 | **0.839** | 0.695 | 1.058 | 0.797 | 1.195 | 0.695 | 1.092 | 0.769 |
| | Avg. | 0.356 | **0.400** | 0.360 | 0.404 | 0.378 | 0.417 | 0.367 | 0.404 | 0.940 | 0.707 | 0.416 | 0.443 | **0.354** | 0.414 | 0.750 | 0.626 | 0.519 | 0.429 | 0.461 | 0.454 |
| Traffic | 96 | 0.455 | 0.291 | **0.395** | **0.268** | 0.649 | 0.389 | 0.462 | 0.295 | 0.522 | 0.290 | 0.593 | 0.321 | 0.650 | 0.396 | 0.788 | 0.499 | 0.587 | 0.366 | 0.612 | 0.338 |
| | 192 | 0.460 | 0.291 | **0.417** | **0.276** | 0.601 | 0.366 | 0.466 | 0.296 | 0.530 | 0.293 | 0.617 | 0.336 | 0.598 | 0.370 | 0.789 | 0.505 | 0.604 | 0.373 | 0.613 | 0.340 |
| | 336 | 0.473 | 0.302 | **0.433** | **0.283** | 0.609 | 0.369 | 0.482 | 0.304 | 0.558 | 0.305 | 0.629 | 0.336 | 0.605 | 0.373 | 0.797 | 0.508 | 0.621 | 0.383 | 0.618 | 0.328 |
| | 720 | 0.500 | 0.321 | **0.467** | **0.302** | 0.647 | 0.387 | 0.514 | 0.322 | 0.589 | 0.328 | 0.640 | 0.350 | 0.645 | 0.394 | 0.841 | 0.523 | 0.626 | 0.382 | 0.653 | 0.355 |
| | Avg. | 0.472 | 0.301 | **0.428** | **0.282** | 0.627 | 0.378 | 0.481 | 0.304 | 0.550 | 0.304 | 0.620 | 0.336 | 0.625 | 0.383 | 0.804 | 0.509 | 0.610 | 0.376 | 0.624 | 0.340 |
| Weather | 96 | 0.159 | **0.206** | 0.174 | 0.214 | 0.192 | 0.232 | 0.177 | 0.218 | **0.158** | 0.230 | 0.172 | 0.220 | 0.196 | 0.255 | 0.221 | 0.306 | 0.217 | 0.296 | 0.173 | 0.223 |
| | 192 | **0.205** | **0.249** | 0.221 | 0.254 | 0.240 | 0.271 | 0.225 | 0.259 | 0.206 | 0.277 | 0.219 | 0.261 | 0.237 | 0.296 | 0.261 | 0.340 | 0.276 | 0.336 | 0.245 | 0.285 |
| | 336 | **0.263** | **0.291** | 0.278 | 0.296 | 0.292 | 0.307 | 0.278 | 0.297 | 0.272 | 0.335 | 0.280 | 0.306 | 0.283 | 0.335 | 0.309 | 0.378 | 0.339 | 0.380 | 0.321 | 0.338 |
| | 720 | **0.344** | **0.345** | 0.358 | 0.347 | 0.364 | 0.353 | 0.354 | 0.348 | 0.398 | 0.418 | 0.365 | 0.359 | 0.345 | 0.381 | 0.377 | 0.427 | 0.403 | 0.428 | 0.414 | 0.410 |
| | Avg. | **0.243** | **0.273** | 0.258 | 0.278 | 0.272 | 0.291 | 0.259 | 0.281 | 0.259 | 0.315 | 0.259 | 0.287 | 0.265 | 0.317 | 0.292 | 0.363 | 0.309 | 0.360 | 0.288 | 0.314 |
| First Count | | **47** | | 19 | | 5 | | 4 | | 1 | | 1 | | 2 | | 0 | | 1 | | 0 | |

Table 1: The results of multivariate long-term time series forecasting. We fix the look-back window length to 96 for all methods, and the forecast horizons are set as $\{96, 192, 336, 720\}$. We highlight the best results in **bold** and the second-best results with underlining. *Avg.* denotes the average results from all four prediction lengths. All reported results are averaged over 5 runs.

| Models | | Ours | Rlinear | PatchTST | Crossformer | FEDformer | TimesNet | DLinear | SCINet | FEDformer | Stationary | Autoformer |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Year | SMAPE | **13.185** | 13.994 | 13.258 | 13.392 | 13.728 | 13.387 | 16.965 | 13.717 | 13.728 | 13.717 | 13.974 |
| | MASE | **2.955** | 3.015 | 2.985 | 3.001 | 3.048 | 2.996 | 4.283 | 3.076 | 3.048 | 3.078 | 3.134 |
| | OWA | **0.775** | 0.807 | 0.781 | 0.787 | 0.803 | 0.786 | 1.058 | 0.807 | 0.803 | 0.807 | 0.822 |
| Quarterly | SMAPE | **9.989** | 10.702 | 10.179 | 16.317 | 10.792 | 10.1 | 12.145 | 10.845 | 10.792 | 10.958 | 11.338 |
| | MASE | **1.17** | 1.299 | 1.212 | 2.197 | 1.283 | 1.182 | 1.52 | 1.295 | 1.283 | 1.325 | 1.365 |
| | OWA | **0.88** | 0.959 | 0.904 | 1.542 | 0.958 | 0.89 | 1.106 | 0.965 | 0.958 | 0.981 | 1.012 |
| Monthly | SMAPE | **12.453** | 13.363 | 12.641 | 12.924 | 14.26 | 12.67 | 13.514 | 13.208 | 14.26 | 13.917 | 13.958 |
| | MASE | **0.913** | 1.014 | 0.93 | 0.966 | 1.102 | 0.933 | 1.037 | 0.999 | 1.102 | 1.097 | 1.103 |
| | OWA | **0.861** | 0.94 | 0.876 | 0.902 | 1.012 | 0.878 | 0.956 | 0.928 | 1.012 | 0.998 | 1.002 |
| Others | SMAPE | **4.565** | 5.437 | 4.946 | 5.439 | 4.954 | 4.891 | 6.709 | 5.432 | 4.954 | 6.302 | 5.458 |
| | MASE | 3.144 | 3.706 | **2.985** | 3.69 | 3.264 | 3.302 | 4.953 | 3.583 | 3.264 | 4.064 | 3.865 |
| | OWA | **0.976** | 1.157 | 1.044 | 1.16 | 1.036 | 1.035 | 1.487 | 1.136 | 1.036 | 1.304 | 1.187 |
| Weighted Average | SMAPE | **11.636** | 12.473 | 11.807 | 13.474 | 12.84 | 11.829 | 13.639 | 12.369 | 12.84 | 12.78 | 12.909 |
| | MASE | **1.556** | 1.677 | 1.59 | 1.866 | 1.701 | 1.585 | 2.095 | 1.677 | 1.701 | 1.756 | 1.771 |
| | OWA | **0.836** | 0.898 | 0.851 | 0.985 | 0.918 | 0.851 | 0.851 | 0.894 | 0.918 | 0.93 | 0.939 |
| First Count | | **14** | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Table 2: The results of multivariate short-term time series forecasting. We highlight the best results in **bold** and the second-best results with underlining. All reported results are averaged over 5 runs. For the details of the metrics, please refer to Appendix D.

only contains one continuous time series, and the samples are obtained by sliding window, while M4 involves 100,000 different time series collected in different frequencies, which makes the forecasting on M4 become challenging (Wu et al.

2023). The metrics we chose adhere (Wu et al. 2023). The **lower** value implies a better performance for all metrics. Please refer to Appendix B for details of the datasets and metrics.

Following (Wu et al. 2023; Liu et al. 2024), we fix the look-back window length to 96, and the forecast horizons are set as $\{96, 192, 336, 720\}$. We include multiple recent transformer-based methods for long-term forecasting, such as PatchTST (Nie et al. 2023), iTransformer (Liu et al. 2024), Crossformer (Zhang and Yan 2023), Fedformer (Zhou et al. 2022), Autoformer (Wu et al. 2021), Non-stationary Transformer (Liu et al. 2022b). We also include recent models with other architecture, such as TimesNet (Wu et al. 2023), RLinear (Li et al. 2023), DLinear (Zeng et al. 2023), and SCINet (Liu et al. 2022a). Since iTransformer is designed for learning cross-variate information, while short-term forecasting is univariate, hence we do not intend to include it in the comparison for short-term forecasting. In the implementation of our method, we use Adam optimizer (Kingma and Ba 2014) and fix the learning rate to 0.0001. The number of the complementors is fixed at 3. Please refer to Appendix C for more details of the implementation. All results are reported over an average of 5 runs. The standard deviation and running time are reported in Appendix D, which confirms the robustness and efficiency of our method.

**Main Results on Long-term Forecasting.** The quantitative results of the long-term time series forecasting are shown in Table 1. The results demonstrate that the proposed model outperforms all other competing methods across most datasets and secures first place 47 times. In most other cases, our method can achieve second place, such as on ECL and Traffic datasets. In comparison, the second-best method (i.e., iTransformer) achieves first place 19 times. Remarkably, Our method shows a significant improvement over PatchTST(2023) and wins it above 42 times across different settings and metrics. All of these results confirm the superiority of our method for long-term time series forecasting. The exemplary qualitative forecasting results are shown in Fig. 4, where our method shows close coherence to the ground truth with minimal error, whereas some transformers (e.g., Non-stationary transformer and Crossformer) show disastrous alignment to the ground truth. We have additional visualization in Appendix D. Please also refer to Appendix D for the results on ILI dataset.

**Main Results on Short-term Forecasting.** The quantitative results for the short-term time series forecasting are shown in Table 2. Different from the long-term forecasting tasks, the short-term forecasting tasks exhibit more temporal variations, as they are collected from different sources. The proposed still demonstrates superior performance on short-term forecasting tasks across different settings compared to other competing methods. Notably, our method secures the best performance 14 times out of all 15 settings. This suggests that the proposed method is even more effective on datasets with more temporal heterogeneity.

## Analysis

**Effectiveness of Sequence Complementors.** We conduct the ablation analysis on Exchange, Weather, ETTm1, and ETTh1 datasets, where the baseline is PatchTST. As shown in Fig. 3, it is evident that adding complementary sequences reduces the forecasting errors. Though there is some variability in performance when using a different number of complementary sequences ($K$), the improvement becomes substantial when $K \geq 3$. Therefore, we fix the number to 3 in our main experiments. For a Wilcoxon signed-rank test, please see Appendix D.

**Effectiveness of Diversification.** We conduct the ablations on the diversification loss when $K \geq 3$, as it is meaningless to diversify only one complementary sequence. As shown in Fig. 3, our model can benefit from this diversification loss with an additional reduction in forecasting error. We provide the similarity matrix for the learned complementary sequences with/without diversification loss in Appendix D.

**Case Study on Training Dynamics.** We conduct a case study on how the richness of the latent representations evolves during training (see Fig. 5). We observe that as training progresses, the richness of the learned representations increases, indicating that the model is learning more complex and diverse features. The curve might plateau once the model reaches a certain level of feature richness, suggesting convergence in feature learning. The main observation is that our model can consistently learn richer representations than the baseline and translate them to a lower forecasting error without slowing the model convergence. This again highlights the effectiveness of complementary sequences.

**Case Study on iTransformer.** We conduct a case study on iTransformer to show that our method can also be applied to other transformers. Though iTransformer has achieved good performance for long-term time series forecasting tasks (see Table 1), its performance is further boosted after integrating the proposed Sequence Complementor (see Table 3). In particular, the integration of Sequence Complementor results in an average reduction of MSE and MAE by 2.4% and 1.22%, respectively. The larger reduction rate of MSE over MAE matches our theoretical analysis that the integration of Sequence Complementor is likely to lower the bound of MSE (Theorem 1). We also observe that the learned representations are enriched after the integration of our method into iTransoformer (for details, see Appendix D). These results suggest that the effectiveness of the proposed method is model-agnostic.

## Related Work

Despite there being other pillars of models in time series forecasting, transformers have been one of the main streams since its introduction. This is because the self-attention in transformers is effective in capturing long-term temporal dependencies and complex multivariate correlations compared to other competing models, such as RNNs (Lai et al. 2018; Patro and Agneeswaran 2024), MLPs (Chen et al. 2023; Wang et al. 2024a), and CNNs (Wu et al. 2023; donghao and wang xue 2024; Liu et al. 2022a; Wu et al. 2023). However, the direct adoption of vanilla transformers faces several challenges in TSF, such as inadequate tokenization of time series sequences, inefficient processing of long sequences, difficulties in series rationalization, and insufficient modeling of variable and channel mutual information. To address these, many

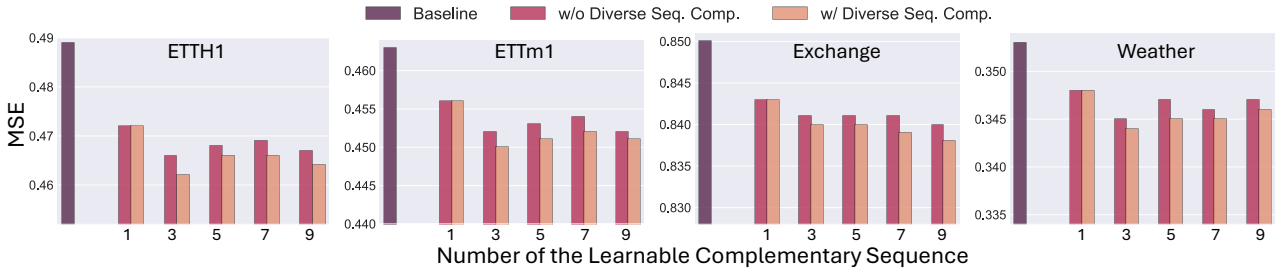



Figure 3: Ablation studies on the number of learnable Sequence Complementors and the diversified Sequence Complementors on different datasets. The results suggest that when the number of complementors is equal to **3**, the overall performance is desired.

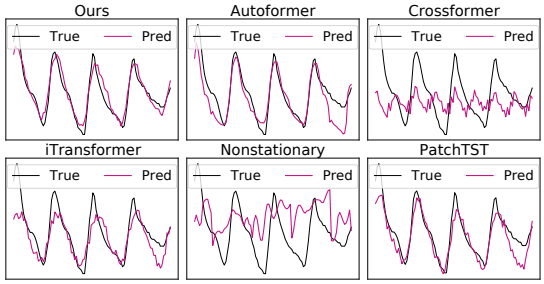



Figure 4: Visualization on ETTh2 Dataset ($T_{out} = 96$).

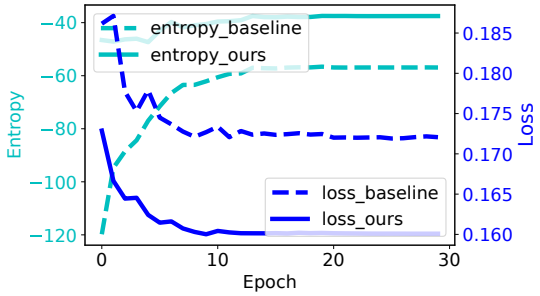



Figure 5: The comparison of training dynamics with and without Sequence Complementors.

| Dataset | ETTm1 | | ETTm2 | | ETTh1 | |
|---|---|---|---|---|---|---|
| Metric | MSE | MAE | MSE | MAE | MSE | MAE |
| iTransformer | 0.407 | 0.41 | 0.288 | 0.332 | 0.454 | 0.447 |
| + Seq. Comp. | **0.393** | **0.402** | **0.285** | **0.329** | **0.443** | **0.441** |
| Reduction (%) | 3.5% | 2.0% | 0.9% | 0.8% | 2.5% | 1.4% |
| Dataset | ETTh2 | | Exchange | | Weather | |
| Metric | MSE | MAE | MSE | MAE | MSE | MAE |
| iTransformer | 0.383 | 0.407 | 0.36 | 0.403 | 0.258 | 0.278 |
| + Seq. Comp. | **0.376** | **0.403** | **0.346** | **0.398** | **0.254** | **0.276** |
| Reduction (%) | 1.9% | 1.1% | 3.8% | 1.3% | 1.7% | 0.7% |

Table 3: The case study on iTransformer, where we add 3 complementary sequences to the patchified features in iTransformer. The performance is averaged over four prediction lengths. Please refer to Appendix D for full results.

follow-up variants have been introduced for TSF. PatchTST (Nie et al. 2023) addresses these problems by segmenting each channel/univariate time sequence into sub-level patchified sequences, designing a tokenization module inspired by ViT (Dosovitskiy et al. 2020). iTransformer (Liu et al. 2024) and Crossformer (Zhang and Yan 2023) further redesign the self-attention and feed-forward network to better capture cross-variate dependencies. Non-stationary Transformer (Liu et al. 2022b) introduces a simple yet effective self-attention for time-series stationarization, enhancing predictive capability for non-stationary series without adding extra parameters. Fedformer (Zhou et al. 2022) and Autoformer (Wu et al. 2021) introduce frequency-enhanced and auto-correlation decomposition into vanilla transformers to efficiently handle the long-range time series with complex patterns.

Although empirical studies such as (Zeng et al. 2023) have questioned the effectiveness of transformers in TSF, we hold the view that self-attention has untapped potentials for time series forecasting due to the modeling of long-range dependencies. Our investigation suggests that the limitations of transformers in TSF stem from their inability to learn rich and generalizable feature representations with limited data. To address this, we propose diversified complementary sequences rather than altering architectural designs, as seen in (Nie et al. 2023; Zhang and Yan 2023; Liu et al. 2024, 2022b; Zhou et al. 2022; Wu et al. 2021). Our method is orthogonal and can be integrated with these previous models.

## Conclusion

In this paper, we investigate the potential reason that leads transformer-based methods with various performances. First, we conduct experiments on multiple recent transformer-based methods and measure the richness of their representation by rank measures and an information-theoretical measure. We show the interesting finding that a richer representation can often translate to a better forecasting performance. Based on this finding and guided by information-theoretical knowledge, we propose the `Sequence Complementors`, which can enhance the representation and be seamlessly integrated into the Transformer-based framework. To further strengthen the complementors, we propose a differentiable volume maximization loss. The empirical results on 8 long-term forecasting datasets and 6 short-term forecasting datasets confirm the superiority of our proposed method. We hope this work provides new insights into understanding and designing transformers for time series forecasting.

## Acknowledgments

This material is based upon the work supported by the National Science Foundation under Grant Number 2204721 and partially supported by our collaborative project with MIT Lincoln Lab under Grant Number 7000612889.

## References

Bengio, Y.; Courville, A.; and Vincent, P. 2013. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8): 1798–1828.

Bhojanapalli, S.; Chakrabarti, A.; Veit, A.; Lukasik, M.; Jain, H.; Liu, F.; Chang, Y.-W.; and Kumar, S. 2021. Leveraging redundancy in attention with reuse transformers. *arXiv preprint arXiv:2110.06821*.

Chen, S.-A.; Li, C.-L.; Arik, S. O.; Yoder, N. C.; and Pfister, T. 2023. TSMixer: An All-MLP Architecture for Time Series Forecast-ing. *Transactions on Machine Learning Research*.

Chen, X.; Li, H.; Amin, R.; and Razi, A. 2024. Learning on Bandwidth Constrained Multi-Source Data with MIMO-inspired DPP MAP Inference. *IEEE Transactions on Machine Learning in Communications and Networking*.

Chen, X.; Li, H.; Qiu, P.; Zhu, W.; Amin, R.; and Razi, A. 2025. Rd-dpp: Rate-distortion theory meets determinantal point process to diversify learning data samples. *IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*.

Dai, Z.; Lai, G.; Yang, Y.; and Le, Q. 2020. Funnel-transformer: Filtering out sequential redundancy for efficient language processing. *Advances in neural information processing systems*, 33: 4271–4282.

Dalvi, F.; Sajjad, H.; Durrani, N.; and Belinkov, Y. 2020. Analyzing redundancy in pretrained transformer models. *arXiv preprint arXiv:2004.04010*.

donghao, L.; and wang xue. 2024. ModernTCN: A Modern Pure Convolution Structure for General Time Series Analysis. In *The Twelfth International Conference on Learning Representations*.

Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

Kingma, D. P.; and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kulesza, A.; Taskar, B.; et al. 2012. Determinantal point processes for machine learning. *Foundations and Trends® in Machine Learning*, 5(2–3): 123–286.

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

LeCun, Y.; Bengio, Y.; and Hinton, G. 2015. Deep learning. *nature*, 521(7553): 436–444.

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

Lin, S.; Lin, W.; Wu, W.; Chen, H.; and Yang, J. 2024. SparseTSF: Modeling Long-term Time Series Forecasting with 1k Parameters. *arXiv preprint arXiv:2405.00946*.

Liu, M.; Zeng, A.; Chen, M.; Xu, Z.; Lai, Q.; Ma, L.; and Xu, Q. 2022a. Scinet: Time series modeling and forecasting with sample convolution and interaction. *Advances in Neural Information Processing Systems*, 35: 5816–5828.

Liu, S.; Yu, H.; Liao, C.; Li, J.; Lin, W.; Liu, A. X.; and Dustdar, S. 2021. Pyraformer: Low-complexity pyramidal attention for long-range time series modeling and forecasting. In *International conference on learning representations*.

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

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

Liu, Z.; Li, J.; Shen, Z.; Huang, G.; Yan, S.; and Zhang, C. 2017. Learning efficient convolutional networks through network slimming. In *Proceedings of the IEEE international conference on computer vision*, 2736–2744.

Ma, Y.; Derksen, H.; Hong, W.; and Wright, J. 2007. Segmentation of multivariate mixed data via lossy data coding and compression. *IEEE transactions on pattern analysis and machine intelligence*, 29(9): 1546–1562.

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

Patro, B. N.; and Agneeswaran, V. S. 2024. Simba: Simplified mamba-based architecture for vision and multivariate time series. *arXiv preprint arXiv:2403.15360*.

Petit, C.; Roumy, A.; and Maugey, T. 2023. A Water-filling Algorithm Maximizing the Volume of Submatrices Above the Rank. In *2023 31st European Signal Processing Conference (EUSIPCO)*, 1295–1299. IEEE.

Sun, C.; Shrivastava, A.; Singh, S.; and Gupta, A. 2017. Revisiting unreasonable effectiveness of data in deep learning era. In *Proceedings of the IEEE international conference on computer vision*, 843–852.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Wang, S.; Wu, H.; Shi, X.; Hu, T.; Luo, H.; Ma, L.; Zhang, J. Y.; and ZHOU, J. 2024a. TimeMixer: Decomposable Multiscale Mixing for Time Series Forecasting. In *The Twelfth International Conference on Learning Representations*.

Wang, Y.; Wu, H.; Dong, J.; Liu, Y.; Long, M.; and Wang, J. 2024b. Deep Time Series Models: A Comprehensive Survey and Benchmark. *arXiv preprint arXiv:2407.13278*.

Wen, W.; Wu, C.; Wang, Y.; Chen, Y.; and Li, H. 2016. Learning structured sparsity in deep neural networks. *Advances in neural information processing systems*, 29.

Wu, H.; Hu, T.; Liu, Y.; Zhou, H.; Wang, J.; and Long, M. 2023. TimesNet: Temporal 2D-Variation Modeling for General Time Series Analysis. In *The Eleventh International Conference on Learning Representations*.

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

Yu, Y.; Chan, K. H. R.; You, C.; Song, C.; and Ma, Y. 2020. Learning diverse and discriminative representations via the principle of maximal coding rate reduction. *Advances in neural information processing systems*, 33: 9422–9434.

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

Zhang, Y.; and Yan, J. 2023. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In *The eleventh international conference on learning representations*.

Zhao, C.; Ni, B.; Zhang, J.; Zhao, Q.; Zhang, W.; and Tian, Q. 2019. Variational convolutional neural network pruning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2780–2789.

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

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