FEDformer: Frequency Enhanced Decomposed Transformer for Long-term Series Forecasting

Tian Zhou [*1] **Ziqing Ma** [*1] **Qingsong Wen** [2] **Xue Wang** [2] **Liang Sun** [2] **Rong Jin** [2]

Abstract

Although Transformer-based methods have significantly improved state-of-the-art results for long-term series forecasting, they are not only computationally expensive but more importantly, are unable to capture the global view of time series (e.g. overall trend). To address these problems, we propose to combine Transformer with the seasonal-trend decomposition method, in which the decomposition method captures the global profile of time series while Transformers capture more detailed structures. To further enhance the performance of Transformer for long-term prediction, we exploit the fact that most time series tend to have a sparse representation in well-known basis such as Fourier transform, and develop a frequency enhanced Transformer. Besides being more effective, the proposed method, termed as Frequency Enhanced Decomposed Transformer (**FEDformer**), is more efficient than standard Transformer with a linear complexity to the sequence length. Our empirical studies with six benchmark datasets show that compared with state-of-the-art methods, FEDformer can reduce prediction error by 14.8% and 22.6% for multivariate and univariate time series, respectively . Code is publicly available at https://github.com/MAZiqing/FEDformer.

1. Introduction

Long-term time series forecasting is a long-standing challenge in various applications (e.g., energy, weather, traffic, economics). Despite the impressive results achieved by RNN-type methods (Rangapuram et al., 2018; Flunkert et al., 2017), they often suffer from the problem of gradient vanishing or exploding (Pascanu et al., 2013), significantly limiting their performance. Following the recent success in NLP and CV community (Vaswani et al., 2017; Devlin et al., 2019; Dosovitskiy et al., 2021; Rao et al., 2021), Transformer (Vaswani et al., 2017) has been introduced to capture long-term dependencies in time series forecasting and shows promising results (Wen et al., 2022; Zhou et al., 2021; Wu et al., 2021). Since high computational complexity and memory requirement make it difficult for Transformer to be applied to long sequence modeling, numerous studies are devoted to reduce the computational cost of Transformer (Li et al., 2019; Kitaev et al., 2020; Zhou et al., 2021; Wang et al., 2020; Xiong et al., 2021; Ma et al., 2021).

Despite the progress made by Transformer-based methods for time series forecasting, they tend to fail in capturing the overall characteristics/distribution of time series in some cases. In Figure 1, we compare the time series of ground truth with that predicted by the vanilla Transformer method (Vaswani et al., 2017) in a real-world ETTm1 dataset (Zhou et al., 2021). It is clear that the predicted time series shared a *different* distribution from that of ground truth. The discrepancy between ground truth and prediction could be explained by the point-wise attention and prediction in Transformer. Since prediction for each timestep is made individually and independently, it is likely that the model fails to maintain the global property and statistics of time series as a whole. To address this problem, we exploit two ideas in this work. The first idea is to incorporate a seasonal-trend decomposition approach (Cleveland et al., 1990; Wen et al., 2019), which is widely used in time series analysis, into the Transformer-based method. Although this idea has been exploited before (Oreshkin et al., 2019; Wu et al., 2021), we present a special design of network that is effective in bringing the distribution of prediction close to that of ground truth, according to Kologrov-Smirnov distribution test. Our second idea is to combine Fourier analysis with the Transformer-based method. Instead of applying Transformer to the time domain, we apply it to the frequency domain which helps Transformer better capture global properties of time series. Combining both ideas, we propose a **F**requency **E**nhanced **D**ecomposition Trans**former**, or,

[*]Equal contribution [1]DAMO Academy, Alibaba Group, Hangzhou, China [2]DAMO Academy, Alibaba Group, Bellevue, USA. Correspondence to: Tian Zhou <tian.zt@alibaba-inc.com>, Rong Jin <jinrong.jr@alibaba-inc.com>.

Proceedings of the 39^{th} International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022.

FEDformer for short, for long-term time series forecasting.

One critical question with FEDformer is which subset of frequency components should be used by Fourier analysis to represent time series. A common wisdom is to keep low-frequency components and throw away the high-frequency ones. This may not be appropriate for time series forecasting as some of trend changes in time series are related to important events, and this piece of information could be lost if we simply remove all high-frequency components. We address this problem by effectively exploiting the fact that time series tend to have (unknown) sparse representations on a basis like Fourier basis. According to our theoretical analysis, a randomly selected subset of frequency components, including both low and high ones, will give a better representation for time series, which is further verified by extensive empirical studies. Besides being more effective for long term forecasting, combining Transformer with frequency analysis allows us to reduce the computational cost of Transformer from quadratic to linear complexity. We note that this is different from previous efforts on speeding up Transformer, which often leads to a performance drop.

In short, we summarize the key contributions of this work as follows:

1. We propose a *frequency enhanced decomposed Transformer* architecture with mixture of experts for seasonal-trend decomposition in order to better capture global properties of time series.
2. We propose *Fourier enhanced blocks* and *Wavelet enhanced blocks* in the Transformer structure that allows us to capture important structures in time series through frequency domain mapping. They serve as substitutions for both self-attention and cross-attention blocks.
3. By randomly selecting a fixed number of Fourier components, the proposed model achieves *linear computational complexity and memory cost*. The effectiveness of this selection method is verified both theoretically and empirically.
4. We conduct extensive experiments over 6 benchmark datasets across multiple domains (energy, traffic, economics, weather and disease). Our empirical studies show that the proposed model improves the performance of state-of-the-art methods by 14.8% and 22.6% for multivariate and univariate forecasting, respectively.

2. Related Work

In this section, an overview of the literature for time series forecasting will be given. The relevant works include traditional times series models (2.1), deep learning models (2.1), Transformer-based models (2.2), and the Fourier Transform in neural networks (2.3).

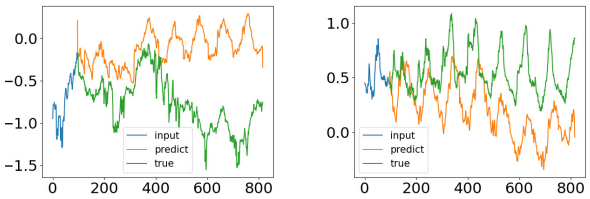

Figure 1. Different distribution between ground truth and forecasting output from vanilla Transformer in a real-world ETTm1 dataset. Left: frequency mode and trend shift. Right: trend shift. xlabel:timestep,ylabel:electrical load value after normalization

2.1. Traditional Time Series Models

Data-driven time series forecasting helps researchers understand the evolution of the systems without architecting the exact physics law behind them. After decades of renovation, time series models have been well developed and served as the backbone of various projects in numerous application fields. The first generation of data-driven methods can date back to 1970. ARIMA (Box & Jenkins, 1968; Box & Pierce, 1970) follows the Markov process and builds an auto-regressive model for recursively sequential forecasting. However, an autoregressive process is not enough to deal with nonlinear and non-stationary sequences. With the bloom of deep neural networks in the new century, recurrent neural networks (RNN) was designed especially for tasks involving sequential data. Among the family of RNNs, LSTM (Hochreiter & Schmidhuber, 1997) and GRU (Chung et al., 2014) employ gated structure to control the information flow to deal with the gradient vanishing or exploration problem. DeepAR (Flunkert et al., 2017) uses a sequential architecture for probabilistic forecasting by incorporating binomial likelihood. Attention based RNN (Qin et al., 2017) uses temporal attention to capture long-range dependencies. However, the recurrent model is not parallelizable and unable to handle long dependencies. The temporal convolutional network (Sen et al., 2019) is another family efficient in sequential tasks. However, limited to the reception field of the kernel, the features extracted still stay local and long-term dependencies are hard to grasp.

2.2. Transformers for Time Series Forecasting

With the innovation of transformers in natural language processing (Vaswani et al., 2017; Devlin et al., 2019) and computer vision tasks (Dosovitskiy et al., 2021; Rao et al., 2021), transformer-based models are also discussed, renovated, and applied in time series forecasting (Zhou et al., 2021; Wu et al., 2021). In sequence to sequence time series forecasting tasks an encoder-decoder architecture is

popularly employed. The self-attention and cross-attention mechanisms are used as the core layers in transformers. However, when employing a point-wise connected matrix, the transformers suffer from quadratic computation complexity.

To get efficient computation without sacrificing too much on performance, the earliest modifications specify the attention matrix with predefined patterns. Examples include: (Qiu et al., 2020) uses block-wise attention which reduces the complexity to the square of block size. Longformer (Beltagy et al., 2020) employs a stride window with fixed intervals. LogTrans (Li et al., 2019) uses log-sparse attention and achieves $N \log^2 N$ complexity. H-transformer (Zhu & Soricut, 2021) uses a hierarchical pattern for sparse approximation of attention matrix with $O(n)$ complexity. Some work uses a combination of patterns (BIGBIRD (Zaheer et al., 2020)) mentioned above. Another strategy is to use dynamic patterns: Reformer (Kitaev et al., 2020) introduces a local-sensitive hashing which reduces the complexity to $N \log N$. (Zhu & Soricut, 2021) introduces a hierarchical pattern. Sinkhorn (Tay et al., 2020) employs a block sorting method to achieve quasi-global attention with only local windows.

Similarly, some work employs a top-k truncating to accelerate computing: Informer (Zhou et al., 2021) uses a KL-divergence based method to select top-k in attention matrix. This sparser matrix costs only $N \log N$ in complexity. Autoformer (Wu et al., 2021) introduces an auto-correlation block in place of canonical attention to get the sub-series level attention, which achieves $N \log N$ complexity with the help of Fast Fourier transform and top-k selection in an auto-correlation matrix.

Another emerging strategy is to employ a low-rank approximation of the attention matrix. Linformer (Wang et al., 2020) uses trainable linear projection to compress the sequence length and achieves $O(n)$ complexity and theoretically proves the boundary of approximation error based on JL lemma. Luna (Ma et al., 2021) develops a nested linear structure with $O(n)$ complexity. Nyströformer (Xiong et al., 2021) leverages the idea of Nyström approximation in the attention mechanism and achieves an $O(n)$ complexity. Performer (Choromanski et al., 2021) adopts an orthogonal random features approach to efficiently model kernelizable attention mechanisms.

2.3. Fourier Transform in Transformers

Thanks to the algorithm of fast Fourier transform (FFT), the computation complexity of Fourier transform is compressed from N^2 to $N \log N$. The Fourier transform has the property that convolution in the time domain is equivalent to multiplication in the frequency domain. Thus the FFT can be used in the acceleration of convolutional networks (Mathieu et al., 2014). FFT can also be used in efficient computing of auto-correlation function (Wen et al., 2021), which can be used as a building neural networks block (Wu et al., 2021) and also useful in numerous time series tasks (Homayouni et al., 2020; Gao et al., 2020).

(Li et al., 2020; Gupta et al., 2021) first introduced Fourier Neural Operator in solving partial differential equations (PDEs). FNO is used as an inner block of networks to perform efficient representation learning in the low-frequency domain. FNO is also proved efficient in computer vision tasks (Rao et al., 2021). It also serves as a working horse to build the Wavelet Neural Operator (WNO), which is recently introduced in solving PEDs (Gupta et al., 2021). While FNO keeps the spectrum modes in low frequency, random Fourier method use randomly selected modes. (Rahimi & Recht, 2008) proposes to map the input data to a randomized low-dimensional feature space to accelerate the training of kernel machines. (Rawat et al., 2019) proposes the Random Fourier softmax (RF-softmax) method that utilizes the powerful Random Fourier Features to enable more efficient and accurate sampling from an approximate softmax distribution. (Lee-Thorp et al., 2021) find that replacing the self-attention layer in a transformer encoder with a standard Fourier Transform achieves much shorter training time with small accuracy loss on the GLUE benchmark.

To the best of our knowledge, our proposed method is the first work to achieve fast attention mechanism through low rank approximated transformation in frequency domain for time series forecasting.

3. Compact Representation of Time Series in Frequency Domain

It is well-known that time series data can be modeled from the time domain and frequency domain. One key contribution of our work which separates from other long-term forecasting algorithms is the frequency-domain operation with a neural network. As Fourier analysis is a common tool to dive into the frequency domain, while how to appropriately represent the information in time series using Fourier analysis is critical. Simply keeping all the frequency components may result in inferior representations since many high-frequency changes in time series are due to noisy inputs. On the other hand, only keeping the low-frequency components may also be inappropriate for series forecasting as some trend changes in time series represent important events. Instead, keeping a compact representation of time series using a small number of selected Fourier components will lead to efficient computation of transformer, which is crucial for modelling long sequences. We propose to represent time series by randomly selecting a constant number of Fourier components, including both high-frequency and low-frequency. Below, an analysis that justifies the random

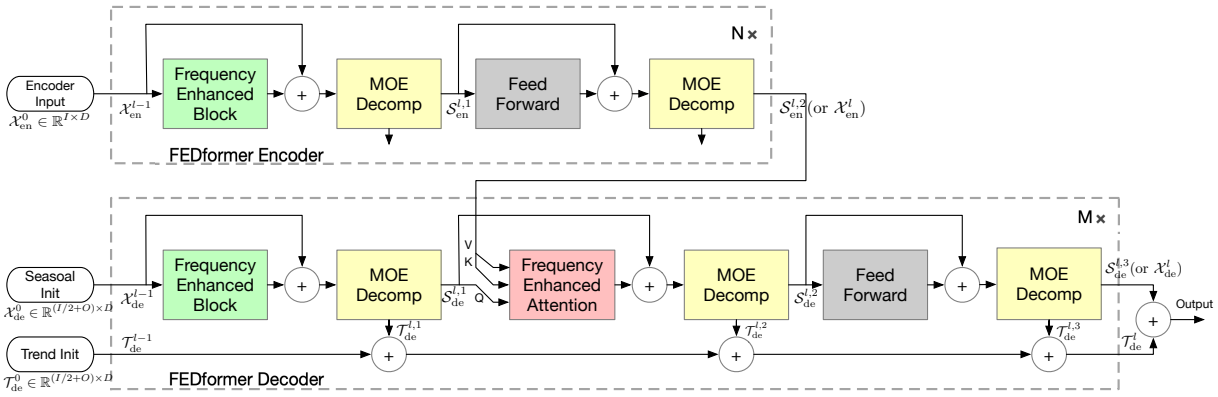

Figure 2. FEDformer Structure. The FEDformer consists of N encoders and M decoders. The Frequency Enhanced Block (FEB, green blocks) and Frequency Enhanced Attention (FEA, red blocks) are used to perform representation learning in frequency domain. Either FEB or FEA has two subversions (FEB-f & FEB-w or FEA-f & FEA-w), where '-f' means using Fourier basis and '-w' means using Wavelet basis. The Mixture Of Expert Decomposition Blocks (MOEDecomp, yellow blocks) are used to extract seasonal-trend patterns from the input data.

selection is presented theoretically. Empirical verification can be found in the experimental session.

Consider we have m time series, denoted as $X_1(t), \ldots, X_m(t)$. By applying Fourier transform to each time series, we turn each $X_i(t)$ into a vector $a_i = (a_{i,1}, \ldots, a_{i,d})^\top \in \mathbb{R}^d$. By putting all the Fourier transform vectors into a matrix, we have $A = (a_1, a_2, \ldots, a_m)^\top \in \mathbb{R}^{m \times d}$, with each row corresponding to a different time series and each column corresponding to a different Fourier component. Although using all the Fourier components allows us to best preserve the history information in the time series, it may potentially lead to overfitting of the history data and consequentially a poor prediction of future signals. Hence, we need to select a subset of Fourier components, that on the one hand should be small enough to avoid the overfitting problem and on the other hand, should be able to preserve most of the history information. Here, we propose to select s components from the d Fourier components ($s < d$) uniformly at random. More specifically, we denote by $i_1 < i_2 < \ldots < i_s$ the randomly selected components. We construct matrix $S \in \{0, 1\}^{s \times d}$, with $S_{i,k} = 1$ if $i = i_k$ and $S_{i,k} = 0$ otherwise. Then, our representation of multivariate time series becomes $A' = AS^\top \in \mathbb{R}^{m \times s}$. Below, we will show that, although the Fourier basis are randomly selected, under a mild condition, A' is able to preserve most of the information from A.

In order to measure how well A' is able to preserve information from A, we project each column vector of A into the subspace spanned by the column vectors in A'. We denote by $P_{A'}(A)$ the resulting matrix after the projection, where $P_{A'}(\cdot)$ represents the projection operator. If A' preserves a large portion of information from A, we would expect a small error between A and $P_{A'}(A)$, i.e. $|A - P_{A'}(A)|$. Let A_k represent the approximation of A by its first k largest single value decomposition. The theorem below shows that $|A - P_{A'}(A)|$ is close to $|A - A_k|$ if the number of randomly sampled Fourier components s is on the order of k^2.

Theorem 1. *Assume that $\mu(A)$, the coherence measure of matrix A, is $\Omega(k/n)$. Then, with a high probability, we have*

$$|A - P_{A'}(A)| \leq (1 + \epsilon)|A - A_k|$$

if $s = O(k^2/\epsilon^2)$.

The detailed analysis can be found in Appendix B.

For real-world multivariate times series, the corresponding matrix A from Fourier transform often exhibit low rank property, since those univaraite variables in multivariate times series depend not only on its past values but also has dependency on each other, as well as share similar frequency components. Therefore, as indicated by the Theorem 1, randomly selecting a subset of Fourier components allows us to appropriately represent the information in Fourier matrix A.

Similarly, wavelet orthogonal polynomials, such as Legendre Polynomials, obey restricted isometry property (RIP) and can be used for capture information in time series as well. Compared to Fourier basis, wavelet based representation is more effective in capturing local structures in time series and thus can be more effective for some forecasting tasks. We defer the discussion of wavelet based representation in Appendix A. In the next section, we will present the design of frequency enhanced decomposed Transformer architecture that incorporate the Fourier transform into transformer.

4. Model Structure

In this section, we will introduce (1) the overall structure of FEDformer, as shown in Figure 2, (2) two subversion structures for signal process: one uses Fourier basis and the other uses Wavelet basis, (3) the mixture of experts mechanism for seasonal-trend decomposition, and (4) the complexity analysis of the proposed model.

4.1. FEDformer Framework

Preliminary Long-term time series forecasting is a sequence to sequence problem. We denote the input length as I and output length as O. We denote D as the hidden states of the series. The input of the encoder is a $I \times D$ matrix and the decoder has $(I/2 + O) \times D$ input.

FEDformer Structure Inspired by the seasonal-trend decomposition and distribution analysis as discussed in Section 1, we renovate Transformer as a deep decomposition architecture as shown in Figure 2, including Frequency Enhanced Block (FEB), Frequency Enhanced Attention (FEA) connecting encoder and decoder, and the Mixture Of Experts Decomposition block (MOEDecomp). The detailed description of FEB, FEA, and MOEDecomp blocks will be given in the following Section 4.2, 4.3, and 4.4 respectively.

The encoder adopts a multilayer structure as: $\mathcal{X}^l_{\text{en}} = \text{Encoder}(\mathcal{X}^{l-1}_{\text{en}})$, where $l \in \{1, \cdots, N\}$ denotes the output of l-th encoder layer and $\mathcal{X}^0_{\text{en}} \in \mathbb{R}^{I \times D}$ is the embedded historical series. The Encoder$(\cdot)$ is formalized as

$$\begin{aligned}
\mathcal{S}^{l,1}_{\text{en}}, _- &= \text{MOEDecomp}(\text{FEB}\left(\mathcal{X}^{l-1}_{\text{en}}\right) + \mathcal{X}^{l-1}_{\text{en}}), \\
\mathcal{S}^{l,2}_{\text{en}}, _- &= \text{MOEDecomp}(\text{FeedForward}\left(\mathcal{S}^{l,1}_{\text{en}}\right) + \mathcal{S}^{l,1}_{\text{en}}), \\
\mathcal{X}^l_{\text{en}} &= \mathcal{S}^{l,2}_{\text{en}},
\end{aligned} \tag{1}$$

where $\mathcal{S}^{l,i}_{\text{en}}, i \in \{1, 2\}$ represents the seasonal component after the i-th decomposition block in the l-th layer respectively. For FEB module, it has two different versions (FEB-f & FEB-w) which are implemented through Discrete Fourier transform (DFT) and Discrete Wavelet transform (DWT) mechanism respectively and can seamlessly replace the self-attention block.

The decoder also adopts a multilayer structure as: $\mathcal{X}^l_{\text{de}}, \mathcal{T}^l_{\text{de}} = \text{Decoder}(\mathcal{X}^{l-1}_{\text{de}}, \mathcal{T}^{l-1}_{\text{de}})$, where $l \in \{1, \cdots, M\}$ denotes the output of l-th decoder layer. The Decoder$(\cdot)$ is formalized as

$$\begin{aligned}
\mathcal{S}^{l,1}_{\text{de}}, \mathcal{T}^{l,1}_{\text{de}} &= \text{MOEDecomp}\left(\text{FEB}\left(\mathcal{X}^{l-1}_{\text{de}}\right) + \mathcal{X}^{l-1}_{\text{de}}\right), \\
\mathcal{S}^{l,2}_{\text{de}}, \mathcal{T}^{l,2}_{\text{de}} &= \text{MOEDecomp}\left(\text{FEA}\left(\mathcal{S}^{l,1}_{\text{de}}, \mathcal{X}^N_{\text{en}}\right) + \mathcal{S}^{l,1}_{\text{de}}\right), \\
\mathcal{S}^{l,3}_{\text{de}}, \mathcal{T}^{l,3}_{\text{de}} &= \text{MOEDecomp}\left(\text{FeedForward}\left(\mathcal{S}^{l,2}_{\text{de}}\right) + \mathcal{S}^{l,2}_{\text{de}}\right), \\
\mathcal{X}^l_{\text{de}} &= \mathcal{S}^{l,3}_{\text{de}}, \\
\mathcal{T}^l_{\text{de}} &= \mathcal{T}^{l-1}_{\text{de}} + \mathcal{W}_{l,1} \cdot \mathcal{T}^{l,1}_{\text{de}} + \mathcal{W}_{l,2} \cdot \mathcal{T}^{l,2}_{\text{de}} + \mathcal{W}_{l,3} \cdot \mathcal{T}^{l,3}_{\text{de}},
\end{aligned} \tag{2}$$

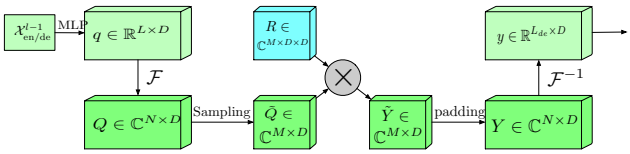

Figure 3. Frequency Enhanced Block with Fourier transform (FEB-f) structure.

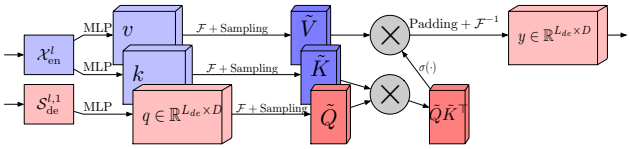

Figure 4. Frequency Enhanced Attention with Fourier transform (FEA-f) structure, $\sigma(\cdot)$ is the activation function.

where $\mathcal{S}^{l,i}_{\text{de}}, \mathcal{T}^{l,i}_{\text{de}}, i \in \{1, 2, 3\}$ represent the seasonal and trend component after the i-th decomposition block in the l-th layer respectively. $\mathcal{W}_{l,i}, i \in \{1, 2, 3\}$ represents the projector for the i-th extracted trend $\mathcal{T}^{l,i}_{\text{de}}$. Similar to FEB, FEA has two different versions (FEA-f & FEA-w) which are implemented through DFT and DWT projection respectively with attention design, and can replace the cross-attention block. The detailed description of FEA$(\cdot)$ will be given in the following Section 4.3.

The final prediction is the sum of the two refined decomposed components as $\mathcal{W}_{\mathcal{S}} \cdot \mathcal{X}^M_{\text{de}} + \mathcal{T}^M_{\text{de}}$, where $\mathcal{W}_{\mathcal{S}}$ is to project the deep transformed seasonal component $\mathcal{X}^M_{\text{de}}$ to the target dimension.

4.2. Fourier Enhanced Structure

Discrete Fourier Transform (DFT) The proposed Fourier Enhanced Structures use discrete Fourier transform (DFT). Let $\mathcal{F}$ denotes the Fourier transform and $\mathcal{F}^{-1}$ denotes the inverse Fourier transform. Given a sequence of real numbers x_n in time domain, where $n = 1, 2...N$. DFT is defined as $X_l = \sum_{n=0}^{N-1} x_n e^{-i\omega ln}$, where i is the imaginary unit and X_l, $l = 1, 2...L$ is a sequence of complex numbers in the frequency domain. Similarly, the inverse DFT is defined as $x_n = \sum_{l=0}^{L-1} X_l e^{i\omega ln}$. The complexity of DFT is $O(N^2)$. With fast Fourier transform (FFT), the computation complexity can be reduced to $O(N \log N)$. Here a random subset of the Fourier basis is used and the scale of the subset is bounded by a scalar. When we choose the mode index before DFT and reverse DFT operations, the computation complexity can be further reduced to $O(N)$.

Frequency Enhanced Block with Fourier Transform (FEB-f) The FEB-f is used in both encoder and decoder as shown in Figure 2. The input ($\boldsymbol{x} \in \mathbb{R}^{N \times D}$) of the FEB-f block is first linearly projected with $\boldsymbol{w} \in \mathbb{R}^{D \times D}$, so $\boldsymbol{q} = \boldsymbol{x} \cdot \boldsymbol{w}$. Then $\boldsymbol{q}$ is converted from the time domain

to the frequency domain. The Fourier transform of $\boldsymbol{q}$ is denoted as $\boldsymbol{Q} \in \mathbb{C}^{N\times D}$. In frequency domain, only the randomly selected M modes are kept so we use a select operator as

$$\tilde{\boldsymbol{Q}} = \text{Select}(\boldsymbol{Q}) = \text{Select}(\mathcal{F}(\boldsymbol{q})), \tag{3}$$

where $\tilde{\boldsymbol{Q}} \in \mathbb{C}^{M\times D}$ and $M << N$. Then, the FEB-f is defined as

$$\text{FEB-f}(\boldsymbol{q}) = \mathcal{F}^{-1}(\text{Padding}(\tilde{\boldsymbol{Q}} \odot \boldsymbol{R})), \tag{4}$$

where $\boldsymbol{R} \in \mathbb{C}^{D\times D\times M}$ is a parameterized kernel initialized randomly. Let $\boldsymbol{Y} = \boldsymbol{Q} \odot \boldsymbol{C}$, with $\boldsymbol{Y} \in \mathbb{C}^{M\times D}$. The production operator $\odot$ is defined as: $Y_{m,d_o} = \sum_{d_i=0}^{D} Q_{m,d_i} \cdot R_{d_i,d_o,m}$, where $d_i = 1, 2...D$ is the input channel and $d_o = 1, 2...D$ is the output channel. The result of $\boldsymbol{Q} \odot \boldsymbol{R}$ is then zero-padded to $\mathbb{C}^{N\times D}$ before performing inverse Fourier transform back to the time domain. The structure is shown in Figure 3.

Frequency Enhanced Attention with Fourier Transform (FEA-f) We use the expression of the canonical transformer. The input: queries, keys, values are denoted as $\boldsymbol{q} \in \mathbb{R}^{L\times D}$, $\boldsymbol{k} \in \mathbb{R}^{L\times D}$, $\boldsymbol{v} \in \mathbb{R}^{L\times D}$. In cross-attention, the queries come from the decoder and can be obtained by $\boldsymbol{q} = \boldsymbol{x}_{en} \cdot \boldsymbol{w}_q$, where $\boldsymbol{w}_q \in \mathbb{R}^{D\times D}$. The keys and values are from the encoder and can be obtained by $\boldsymbol{k} = \boldsymbol{x}_{de} \cdot \boldsymbol{w}_k$ and $\boldsymbol{v} = \boldsymbol{x}_{de} \cdot \boldsymbol{w}_v$, where $\boldsymbol{w}_k, \boldsymbol{w}_v \in \mathbb{R}^{D\times D}$. Formally, the canonical attention can be written as

$$\text{Atten}(\boldsymbol{q}, \boldsymbol{k}, \boldsymbol{v}) = \text{Softmax}(\frac{\boldsymbol{q}\boldsymbol{k}^\top}{\sqrt{d_q}})\boldsymbol{v}. \tag{5}$$

In FEA-f, we convert the queries, keys, and values with Fourier Transform and perform a similar attention mechanism in the frequency domain, by randomly selecting M modes. We denote the selected version after Fourier Transform as $\tilde{\boldsymbol{Q}} \in \mathbb{C}^{M\times D}$, $\tilde{\boldsymbol{K}} \in \mathbb{C}^{M\times D}$, $\tilde{\boldsymbol{V}} \in \mathbb{C}^{M\times D}$. The FEA-f is defined as

$$\begin{aligned} \tilde{\boldsymbol{Q}} &= \text{Select}(\mathcal{F}(\boldsymbol{q})) \\ \tilde{\boldsymbol{K}} &= \text{Select}(\mathcal{F}(\boldsymbol{k})) \\ \tilde{\boldsymbol{V}} &= \text{Select}(\mathcal{F}(\boldsymbol{v})) \end{aligned} \tag{6}$$

$$\text{FEA-f}(\boldsymbol{q}, \boldsymbol{k}, \boldsymbol{v}) = \mathcal{F}^{-1}(\text{Padding}(\sigma(\tilde{\boldsymbol{Q}} \cdot \tilde{\boldsymbol{K}}^\top) \cdot \tilde{\boldsymbol{V}})), \tag{7}$$

where σ is the activation function. We use softmax or tanh for activation, since their converging performance differs in different data sets. Let $\boldsymbol{Y} = \sigma(\tilde{\boldsymbol{Q}} \cdot \tilde{\boldsymbol{K}}^\top) \cdot \tilde{\boldsymbol{V}}$, and $\boldsymbol{Y} \in \mathbb{C}^{M\times D}$ needs to be zero-padded to $\mathbb{C}^{L\times D}$ before performing inverse Fourier transform. The FEA-f structure is shown in Figure 4.

4.3. Wavelet Enhanced Structure

Discrete Wavelet Transform (DWT) While the Fourier transform creates a representation of the signal in the frequency domain, the Wavelet transform creates a representation in both the frequency and time domain, allowing efficient access of localized information of the signal. The multiwavelet transform synergizes the advantages of orthogonal polynomials as well as wavelets. More designed detail is described in Appendix C.

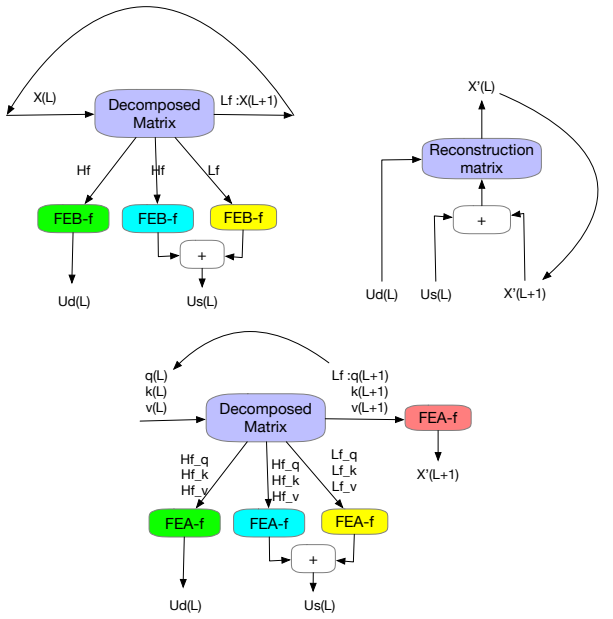

Figure 5. Top Left: Wavelet frequency enhanced block decomposition stage. Top Right: Wavelet block reconstruction stage shared by FEB-w and FEA-w. Bottom: Wavelet frequency enhanced cross attention decomposition stage.

Frequency Enhanced Block with Wavelet Transform (FEB-w) The overall FEB-w architecture is shown in Figure 5. It differs from FEB-f in the recursive mechanism: the input is decomposed into 3 parts recursively and operates individually. For the wavelet decomposition part, we implement the fixed Legendre wavelets basis decomposition matrix. Three FEB-f modules are used to process the resulting high-frequency part, low-frequency part, and remaining part from wavelet decomposition respectively.

Frequency Enhanced Attention with Wavelet Transform (FEA-w) FEA-w contains the decomposition stage and reconstruction stage like FEB-w. Here we keep the reconstruction stage unchanged. The only difference lies in the decomposition stage. The same decomposed matrix is used to decompose $\boldsymbol{q}, \boldsymbol{k}, \boldsymbol{v}$ signal separately, and $\boldsymbol{q}, \boldsymbol{k}, \boldsymbol{v}$ share the same sets of module to process them as well. As shown above, a frequency enhanced block with wavelet decomposition block (FEB-w) contains three FEB-f blocks for the signal process. We can view the FEB-f as a substitution of self-attention mechanism. We use a straightforward way to build the frequency enhanced cross attention with wavelet decomposition, substituting each FEB-f with a FEA-f module. Besides, another FEA-f module is added to process the

Table 1. Complexity analysis of different forecasting models.

Methods	Training		Testing
	Time	Memory	Steps
FEDformer	$\mathcal{O}(L)$	$\mathcal{O}(L)$	1
Autoformer	$\mathcal{O}(L \log L)$	$\mathcal{O}(L \log L)$	1
Informer	$\mathcal{O}(L \log L)$	$\mathcal{O}(L \log L)$	1
Transformer	$\mathcal{O}\left(L^2\right)$	$\mathcal{O}\left(L^2\right)$	L
LogTrans	$\mathcal{O}(L \log L)$	$\mathcal{O}\left(L^2\right)$	1
Reformer	$\mathcal{O}(L \log L)$	$\mathcal{O}(L \log L)$	L
LSTM	$\mathcal{O}(L)$	$\mathcal{O}(L)$	L

coarsest remaining $q(L), k(L), v(L)$ signal.

4.4. Mixture of Experts for Seasonal-Trend Decomposition

Because of the commonly observed complex periodic pattern coupled with the trend component on real-world data, extracting the trend can be hard with fixed window average pooling. To overcome such a problem, we design a Mixture Of Experts Decomposition block (MOEDecomp). It contains a set of average filters with different sizes to extract multiple trend components from the input signal and a set of data-dependent weights for combining them as the final trend. Formally, we have

$$\mathbf{X}_{\mathbf{trend}} = \mathbf{Softmax}(L(x)) * (F(x)), \tag{8}$$

where $F(\cdot)$ is a set of average pooling filters and Softmax$(L(x))$ is the weights for mixing these extracted trends.

4.5. Complexity Analysis

For FEDformer-f, the computational complexity for time and memory is $O(L)$ with a fixed number of randomly selected modes in FEB & FEA blocks. We set modes number $M = 64$ as default value. Though the complexity of full DFT transformation by FFT is $(O(L \log(L))$, our model only needs $O(L)$ cost and memory complexity with the preselected set of Fourier basis for quick implementation. For FEDformer-w, when we set the recursive decompose step to a fixed number L and use a fixed number of randomly selected modes the same as FEDformer-f, the time complexity and memory usage are $O(L)$ as well. In practice, we choose $L = 3$ and modes number $M = 64$ as default value. The comparisons of the time complexity and memory usage in training and the inference steps in testing are summarized in Table 1. It can be seen that the proposed FEDformer achieves the best overall complexity among Transformer-based forecasting models.

5. Experiments

To evaluate the proposed FEDformer, we conduct extensive experiments on six popular real-world datasets, including energy, economics, traffic, weather, and disease. Since classic models like ARIMA and basic RNN/CNN models perform relatively inferior as shown in (Zhou et al., 2021) and (Wu et al., 2021), we mainly include four state-of-the-art transformer-based models for comparison, i.e., Autoformer (Wu et al., 2021), Informer (Zhou et al., 2021), LogTrans (Li et al., 2019) and Reformer (Kitaev et al., 2020) as baseline models. Note that since Autoformer holds the best performance in all the six benchmarks, it is used as the main baseline model for comparison. More details about baseline models, datasets, and implementation are described in Appendix 2.2, E.1, and E.2, respectively.

5.1. Main Results

For better comparison, we follow the experiment settings of Autoformer in (Wu et al., 2021) where the input length is fixed to 96, and the prediction lengths for both training and evaluation are fixed to be 96, 192, 336, and 720, respectively.

Multivariate Results For the multivariate forecasting, FEDformer achieves the best performance on all six benchmark datasets at all horizons as shown in Table 2. Compared with Autoformer, the proposed FEDformer yields an overall **14.8%** relative MSE reduction. It is worth noting that for some of the datasets, such as Exchange and ILI, the improvement is even more significant (over 20%). Note that the Exchange dataset does not exhibit clear periodicity in its time series, but FEDformer can still achieve superior performance. Overall, the improvement made by FEDformer is consistent with varying horizons, implying its strength in long term forecasting. More detailed results on ETT full benchmark are provided in Appendix E.3.

Univariate Results The results for univariate time series forecasting are summarized in Table 3. Compared with Autoformer, FEDformer yields an overall **22.6%** relative MSE reduction, and on some datasets, such as traffic, the improvement can be more than 30%. It again verifies that FEDformer is more effective in long-term forecasting. Note that due to the difference between Fourier and wavelet basis, FEDformer-f and FEDformer-w perform well on different datasets, making them complementary choice for long term forecasting. More detailed results on ETT full benchmark are provided in Appendix E.3.

5.2. Ablation Studies

In this section, the ablation experiments are conducted, aiming at comparing the performance of frequency enhanced block and its alternatives. The current SOTA results of Autoformer which uses the autocorrelation mechanism serve as the baseline. Three ablation variants of FEDformer are tested: 1) FEDformer V1: we use FEB to substitute self-attention only; 2) FEDformer V2: we use FEA to substitute

Table 2. Multivariate long-term series forecasting results on six datasets with input length $I = 96$ and prediction length $O \in \{96, 192, 336, 720\}$ (For ILI dataset, we use input length $I = 36$ and prediction length $O \in \{24, 36, 48, 60\}$). A lower MSE indicates better performance, and the best results are highlighted in bold.

Methods	Metric	ETTm2				Electricity				Exchange				Traffic				Weather				ILI			
		96	192	336	720	96	192	336	720	96	192	336	720	96	192	336	720	96	192	336	720	24	36	48	60
FEDformer-f	MSE	**0.203**	**0.269**	**0.325**	**0.421**	0.193	0.201	0.214	0.246	0.148	0.271	0.460	1.195	0.587	0.604	0.621	0.626	**0.217**	**0.276**	**0.339**	**0.403**	3.228	2.679	2.622	2.857
	MAE	**0.287**	**0.328**	**0.366**	**0.415**	0.308	0.315	0.329	0.355	0.278	0.380	0.500	0.841	0.366	0.373	0.383	0.382	**0.296**	**0.336**	**0.380**	**0.428**	1.260	1.080	1.078	1.157
FEDformer-w	MSE	0.204	0.316	0.359	0.433	**0.183**	**0.195**	**0.212**	**0.231**	**0.139**	**0.256**	**0.426**	**1.090**	**0.562**	**0.562**	**0.570**	**0.596**	0.227	0.295	0.381	0.424	**2.203**	**2.272**	**2.209**	**2.545**
	MAE	0.288	0.363	0.387	0.432	**0.297**	**0.308**	**0.313**	**0.343**	**0.276**	**0.369**	**0.464**	**0.800**	**0.349**	**0.346**	**0.323**	**0.368**	0.304	0.363	0.416	0.434	**0.963**	**0.976**	**0.981**	**1.061**
Autoformer	MSE	0.255	0.281	0.339	0.422	0.201	0.222	0.231	0.254	0.197	0.300	0.509	1.447	0.613	0.616	0.622	0.660	0.266	0.307	0.359	0.419	3.483	3.103	2.669	2.770
	MAE	0.339	0.340	0.372	0.419	0.317	0.334	0.338	0.361	0.323	0.369	0.524	0.941	0.388	0.382	0.337	0.408	0.336	0.367	0.395	0.428	1.287	1.148	1.085	1.125
Informer	MSE	0.365	0.533	1.363	3.379	0.274	0.296	0.300	0.373	0.847	1.204	1.672	2.478	0.719	0.696	0.777	0.864	0.300	0.598	0.578	1.059	5.764	4.755	4.763	5.264
	MAE	0.453	0.563	0.887	1.338	0.368	0.386	0.394	0.439	0.752	0.895	1.036	1.310	0.391	0.379	0.420	0.472	0.384	0.544	0.523	0.741	1.677	1.467	1.469	1.564
LogTrans	MSE	0.768	0.989	1.334	3.048	0.258	0.266	0.280	0.283	0.968	1.040	1.659	1.941	0.684	0.685	0.7337	0.717	0.458	0.658	0.797	0.869	4.480	4.799	4.800	5.278
	MAE	0.642	0.757	0.872	1.328	0.357	0.368	0.380	0.376	0.812	0.851	1.081	1.127	0.384	0.390	0.408	0.396	0.490	0.589	0.652	0.675	1.444	1.467	1.468	1.560
Reformer	MSE	0.658	1.078	1.549	2.631	0.312	0.348	0.350	0.340	1.065	1.188	1.357	1.510	0.732	0.733	0.742	0.755	0.689	0.752	0.639	1.130	4.400	4.783	4.832	4.882
	MAE	0.619	0.827	0.972	1.242	0.402	0.433	0.433	0.420	0.829	0.906	0.976	1.016	0.423	0.420	0.420	423	0.596	0.638	0.596	0.792	1.382	1.448	1.465	1.483

Table 3. Univariate long-term series forecasting results on six datasets with input length $I = 96$ and prediction length $O \in \{96, 192, 336, 720\}$ (For ILI dataset, we use input length $I = 36$ and prediction length $O \in \{24, 36, 48, 60\}$). A lower MSE indicates better performance, and the best results are highlighted in bold.

Methods	Metric	ETTm2				Electricity				Exchange				Traffic				Weather				ILI			
		96	192	336	720	96	192	336	720	96	192	336	720	96	192	336	720	96	192	336	720	24	36	48	60
FEDformer-f	MSE	0.072	**0.102**	**0.130**	**0.178**	**0.253**	**0.282**	0.346	**0.422**	0.154	0.286	0.511	1.301	0.207	0.205	0.219	0.244	0.0062	0.0060	0.0041	0.0055	0.708	0.584	0.717	0.855
	MAE	0.206	**0.245**	**0.279**	**0.325**	0.370	**0.386**	0.431	**0.484**	0.304	0.420	0.555	0.879	0.312	0.312	0.323	0.344	0.062	0.062	0.050	0.059	0.627	0.617	0.697	0.774
FEDformer-w	MSE	**0.063**	0.110	0.147	0.219	0.262	0.316	0.361	0.448	**0.131**	**0.277**	**0.426**	**1.162**	**0.170**	**0.173**	**0.178**	**0.187**	**0.0035**	0.0054	0.008	0.015	**0.693**	**0.554**	**0.699**	**0.828**
	MAE	**0.189**	0.252	0.301	0.368	0.378	0.410	0.445	0.501	**0.284**	**0.420**	**0.511**	**0.832**	**0.263**	**0.265**	**0.266**	**0.286**	0.046	0.059	0.072	0.091	**0.629**	**0.604**	**0.696**	**0.770**
Autoformer	MSE	0.065	0.118	0.154	0.182	0.341	0.345	0.406	0.565	0.241	0.300	0.509	1.260	0.246	0.266	0.263	0.269	0.011	0.0075	0.0063	0.0085	0.948	0.634	0.791	0.874
	MAE	0.189	0.256	0.305	0.335	0.438	0.428	0.470	0.581	0.387	0.369	0.524	0.867	0.346	0.370	0.371	0.372	0.081	0.067	0.062	0.070	0.732	0.650	0.752	0.797
Informer	MSE	0.080	0.112	0.166	0.228	0.258	0.285	**0.336**	0.607	1.327	1.258	2.179	1.280	0.257	0.299	0.312	0.366	0.004	**0.002**	**0.004**	**0.003**	5.282	4.554	4.273	5.214
	MAE	0.217	0.259	0.314	0.380	**0.367**	0.388	**0.423**	0.599	0.944	0.924	1.296	0.953	0.353	0.376	0.387	0.436	**0.044**	**0.040**	**0.049**	**0.042**	2.050	1.916	1.846	2.057
LogTrans	MSE	0.075	0.129	0.154	0.160	0.288	0.432	0.430	0.491	0.237	0.738	2.018	2.405	0.226	0.314	0.387	0.437	0.0046	0.0060	0.0060	0.007	3.607	2.407	3.106	3.698
	MAE	0.208	0.275	0.302	0.322	0.393	0.483	0.483	0.531	0.377	0.619	1.070	1.175	0.317	0.408	0.453	0.491	0.052	0.060	0.054	0.059	1.662	1.363	1.575	1.733
Reformer	MSE	0.077	0.138	0.160	0.168	0.275	0.304	0.370	0.460	0.298	0.777	1.833	1.203	0.313	0.386	0.423	0.378	0.012	0.0098	0.013	0.011	3.838	2.934	3.755	4.162
	MAE	0.214	0.290	0.313	0.334	0.379	0.402	0.448	0.511	0.444	0.719	1.128	0.956	0.383	0.453	0.468	0.433	0.087	0.044	0.100	0.083	1.720	1.520	1.749	1.847

cross attention only; 3) FEDFormer V3: we use FEA to substitute both self and cross attention. The ablated versions of FEDformer-f as well as the SOTA models are compared in Table 4, and we use a bold number if the ablated version brings improvements compared with Autoformer. We omit the similar results in FEDformer-w due to space limit. It can be seen in Table 4 that FEDformer V1 brings improvement in 10/16 cases, while FEDformer V2 improves in 12/16 cases. The best performance is achieved in our FEDformer with FEB and FEA blocks which improves performance in all 16/16 cases. This verifies the effectiveness of the designed FEB, FEA for substituting self and cross attention. Furthermore, experiments on ETT and Weather datasets show that the adopted MOEDecomp (mixture of experts decomposition) scheme can bring an average of 2.96% improvement compared with the single decomposition scheme. More details are provided in Appendix E.5.

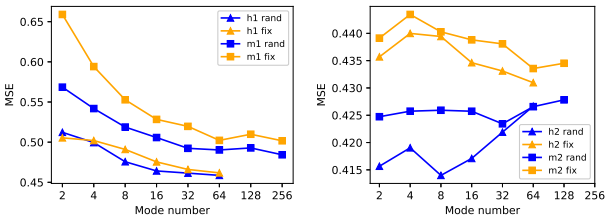

Figure 6. Comparison of two base-modes selection method (Fix&Rand). Rand policy means randomly selecting a subset of modes, Fix policy means selecting the lowest frequency modes. Two policies are compared on a variety of base-modes number $M \in \{2, 4, 8...256\}$ on ETT full-benchmark (h1, m1, h2, m2).

5.3. Mode Selection Policy

The selection of discrete Fourier basis is the key to effectively representing the signal and maintaining the model's **linear** complexity. As we discussed in Section 3, random Fourier mode selection is a better policy in forecasting tasks. more importantly, random policy requires no prior knowledge of the input and generalizes easily in new tasks. Here we empirically compare the random selection policy with fixed selection policy, and summarize the experimental results in Figure 6. It can be observed that the adopted random policy achieves better performance than the common fixed policy which only keeps the low frequency modes. Meanwhile, the random policy exhibits some **mode saturation effect**, indicating an appropriate random number of modes instead of all modes would bring better performance, which is also consistent with the theoretical analysis in Section 3.

Table 4. Ablation studies: multivariate long-term series forecasting results on ETTm1 and ETTm2 with input length $I = 96$ and prediction length $O \in \{96, 192, 336, 720\}$. Three variants of FEDformer-f are compared with baselines. The best results are highlighted in bold.

Methods		Transformer		Informer		Autoformer		FEDformer V1		FEDformer V2		FEDformer V3		FEDformer-f	
Self-att		FullAtt		ProbAtt		AutoCorr		FEB-f(Eq. 4)		AutoCorr		FEA-f(Eq. 7)		FEB-f(Eq. 4)	
Cross-att		FullAtt		ProbAtt		AutoCorr		AutoCorr		FEA-f(Eq. 7)		FEA-f(Eq. 7)		FEA-f(Eq. 7)	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTm1	96	0.525	0.486	0.458	0.465	0.481	0.463	**0.378**	0.419	0.539	0.490	0.534	0.482	0.379	**0.419**
	192	0.526	0.502	0.564	0.521	0.628	0.526	**0.417**	0.442	0.556	0.499	0.552	0.493	0.426	**0.441**
	336	0.514	0.502	0.672	0.559	0.728	0.567	0.480	0.477	0.541	0.498	0.565	0.503	**0.445**	**0.459**
	720	0.564	0.529	0.714	0.596	0.658	0.548	0.543	0.517	0.558	0.507	0.585	0.515	**0.543**	**0.490**
ETTm2	96	0.268	0.346	0.227	0.305	0.255	0.339	0.259	0.337	0.216	0.297	0.211	0.292	**0.203**	**0.287**
	192	0.304	0.355	0.300	0.360	0.281	0.340	0.285	0.344	0.274	0.331	0.272	0.329	**0.269**	**0.328**
	336	0.365	0.400	0.382	0.410	0.339	0.372	**0.320**	0.373	0.334	0.369	0.327	**0.363**	0.325	0.366
	720	0.475	0.466	1.637	0.794	0.422	0.419	0.761	0.628	0.427	0.420	**0.418**	0.415	0.421	**0.415**
Count		0	0	0	0	0	0	3	1	0	0	1	1	4	7

5.4. Distribution Analysis of Forecasting Output

In this section, we evaluate the distribution similarity between the input sequence and forecasting output of different transformer models quantitatively. In Table 5, we applied the Kolmogrov-Smirnov test to check if the forecasting results of different models made on ETTm1 and ETTm2 are consistent with the input sequences. In particular, we test if the input sequence of fixed 96-time steps come from the same distribution as the predicted sequence, with the null hypothesis that both sequences come from the same distribution. On both datasets, by setting the common P-value as 0.01, various existing Transformer baseline models have much less values than 0.01 except Autoformer, which indicates their forecasting output have a higher probability to be sampled from the different distributions compared to the input sequence. In contrast, Autoformer and FEDformer have much larger P-value compared to others, which mainly contributes to their seasonal-trend decomposition mechanism. Though we get close results from ETTm2 by both models, the proposed FEDformer has much larger P-value in ETTm1. And it's the only model whose null hypothesis can not be rejected with P-value larger than 0.01 in all cases of the two datasets, implying that the output sequence generated by FEDformer shares a more similar distribution as the input sequence than others and thus justifies the our design motivation of FEDformer as discussed in Section 1. More detailed analysis are provided in Appendix D.

5.5. Differences Compared to Autoformer baseline

Since we use the decomposed encoder-decoder overall architecture as Autoformer, we think it is critical to emphasize the differences. In Autoformer, the authors consider a nice idea to use the top-k sub-sequence correlation (autocorrelation) module instead of point-wise attention, and the Fourier method is applied to improve the efficiency for sub-sequence level similarity computation. In general, Autoformer can be considered as decomposing the sequence into multiple time domain sub-sequences for feature exaction.

Table 5. P-values of Kolmogrov-Smirnov test of different transformer models for long-term forecasting output on ETTm1 and ETTm2 dataset. Larger value indicates the hypothesis (the input sequence and forecasting output come from the same distribution) is less likely to be rejected. The best results are highlighted.

Methods		Transformer	Informer	Autoformer	FEDformer	True
ETTm1	96	0.0090	0.0055	0.020	**0.048**	0.023
	192	0.0052	0.0029	0.015	**0.028**	0.013
	336	0.0022	0.0019	0.012	**0.015**	0.010
	720	0.0023	0.0016	0.008	**0.014**	0.004
ETTm2	96	0.0012	0.0008	**0.079**	0.071	0.087
	192	0.0011	0.0006	**0.047**	0.045	0.060
	336	0.0005	0.00009	0.027	**0.028**	0.042
	720	0.0008	0.0002	**0.023**	0.021	0.023
Count		0	0	3	5	NA

In contrast, We use frequency transform to decompose the sequence into multiple frequency domain modes to extract the feature. In particular, we do not use a selective approach in sub-sequence selection. Instead, all frequency features are computed from the whole sequence, and this global property makes our model engage better performance for long sequence.

6. Conclusions

This paper proposes a frequency enhanced transformer model for long-term series forecasting which achieves state-of-the-art performance and enjoys linear computational complexity and memory cost. We propose an attention mechanism with low-rank approximation in frequency and a mixture of experts decomposition to control the distribution shifting. The proposed frequency enhanced structure decouples the input sequence length and the attention matrix dimension, leading to the linear complexity. Moreover, we theoretically and empirically prove the effectiveness of the adopted random mode selection policy in frequency. Lastly, extensive experiments show that the proposed model achieves the best forecasting performance on six benchmark datasets in comparison with four state-of-the-art algorithms.

References

Arriaga, R. I. and Vempala, S. S. An algorithmic theory of learning: Robust concepts and random projection. *Mach. Learn.*, 63(2):161–182, 2006.

Beltagy, I., Peters, M. E., and Cohan, A. Longformer: The long-document transformer. *CoRR*, abs/2004.05150, 2020.

Box, G. E. P. and Jenkins, G. M. Some recent advances in forecasting and control. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 17(2):91–109, 1968.

Box, G. E. P. and Pierce, D. A. Distribution of residual autocorrelations in autoregressive-integrated moving average time series models. volume 65, pp. 1509–1526. Taylor & Francis, 1970.

Choromanski, K. M., Likhosherstov, V., Dohan, D., Song, X., Gane, A., Sarlós, T., Hawkins, P., Davis, J. Q., Mohiuddin, A., Kaiser, L., Belanger, D. B., Colwell, L. J., and Weller, A. Rethinking attention with performers. In *9th International Conference on Learning Representations (ICLR), Virtual Event, Austria, May 3-7, 2021*, 2021.

Chung, J., Gülçehre, Ç., Cho, K., and Bengio, Y. Empirical evaluation of gated recurrent neural networks on sequence modeling. *CoRR*, abs/1412.3555, 2014.

Cleveland, R. B., Cleveland, W. S., McRae, J. E., and Terpenning, I. Stl: A seasonal-trend decomposition. *Journal of official statistics*, 6(1):3–73, 1990.

Devlin, J., Chang, M., Lee, K., and Toutanova, K. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT), Minneapolis, MN, USA, June 2-7, 2019*, pp. 4171–4186, 2019.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. An image is worth 16x16 words: Transformers for image recognition at scale. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

Drineas, P., Mahoney, M. W., and Muthukrishnan, S. Relative-error CUR matrix decompositions. *CoRR*, abs/0708.3696, 2007.

Flunkert, V., Salinas, D., and Gasthaus, J. Deepar: Probabilistic forecasting with autoregressive recurrent networks. *CoRR*, abs/1704.04110, 2017.

Gao, J., Song, X., Wen, Q., Wang, P., Sun, L., and Xu, H. Robusttad: Robust time series anomaly detection via decomposition and convolutional neural networks. *KDD Workshop on Mining and Learning from Time Series (KDD-MileTS'20)*, 2020.

Gupta, G., Xiao, X., and Bogdan, P. Multiwavelet-based operator learning for differential equations, 2021.

Hochreiter, S. and Schmidhuber, J. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, November 1997. ISSN 0899-7667, 1530-888X.

Homayouni, H., Ghosh, S., Ray, I., Gondalia, S., Duggan, J., and Kahn, M. G. An autocorrelation-based lstm-autoencoder for anomaly detection on time-series data. In *2020 IEEE International Conference on Big Data (Big Data)*, pp. 5068–5077, 2020.

Johnson, W. B. Extensions of lipschitz mappings into hilbert space. *Contemporary mathematics*, 26:189–206, 1984.

Kingma, D. P. and Ba, J. Adam: A Method for Stochastic Optimization. *arXiv:1412.6980 [cs]*, January 2017. arXiv: 1412.6980.

Kitaev, N., Kaiser, L., and Levskaya, A. Reformer: The efficient transformer. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*, 2020.

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.

Lee-Thorp, J., Ainslie, J., Eckstein, I., and Ontañón, S. FNet: Mixing tokens with fourier transforms. *CoRR*, abs/2105.03824, 2021.

Li, S., Jin, X., Xuan, Y., Zhou, X., Chen, W., Wang, Y.-X., and Yan, X. Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Li, Z., Kovachki, N. B., Azizzadenesheli, K., Liu, B., Bhattacharya, K., Stuart, A. M., and Anandkumar, A. Fourier neural operator for parametric partial differential equations. *CoRR*, abs/2010.08895, 2020.

Ma, X., Kong, X., Wang, S., Zhou, C., May, J., Ma, H., and Zettlemoyer, L. Luna: Linear unified nested attention. *CoRR*, abs/2106.01540, 2021.

Mathieu, M., Henaff, M., and LeCun, Y. Fast training of convolutional networks through ffts. In *2nd International*

Conference on Learning Representations (ICLR), Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings, 2014.

Oreshkin, B. N., Carpov, D., Chapados, N., and Bengio, Y. N-BEATS: Neural basis expansion analysis for interpretable time series forecasting. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2019.

Pascanu, R., Mikolov, T., and Bengio, Y. On the difficulty of training recurrent neural networks. In *Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Atlanta, GA, USA, 16-21 June 2013*, volume 28, pp. 1310–1318, 2013.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, pp. 8024–8035. 2019.

Qin, Y., Song, D., Chen, H., Cheng, W., Jiang, G., and Cottrell, G. W. A dual-stage attention-based recurrent neural network for time series prediction. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI), Melbourne, Australia, August 19-25, 2017*, pp. 2627–2633. ijcai.org, 2017.

Qiu, J., Ma, H., Levy, O., Yih, W., Wang, S., and Tang, J. Blockwise self-attention for long document understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2020, Online Event, 16-20 November 2020*, volume EMNLP 2020 of *Findings of ACL*, pp. 2555–2565. Association for Computational Linguistics, 2020.

Rahimi, A. and Recht, B. Random features for large-scale kernel machines. In Platt, J., Koller, D., Singer, Y., and Roweis, S. (eds.), *Advances in Neural Information Processing Systems*, volume 20. Curran Associates, Inc., 2008.

Rangapuram, S. S., Seeger, M. W., Gasthaus, J., Stella, L., Wang, Y., and Januschowski, T. Deep state space models for time series forecasting. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

Rao, Y., Zhao, W., Zhu, Z., Lu, J., and Zhou, J. Global filter networks for image classification. *CoRR*, abs/2107.00645, 2021.

Rawat, A. S., Chen, J., Yu, F. X., Suresh, A. T., and Kumar, S. Sampled softmax with random fourier features. In *Advances in Neural Information Processing Systems (NeurIPS), December 8-14, 2019, Vancouver, BC, Canada*, pp. 13834–13844, 2019.

Sen, R., Yu, H., and Dhillon, I. S. Think globally, act locally: A deep neural network approach to high-dimensional time series forecasting. In *Advances in Neural Information Processing Systems (NeurIPS), December 8-14, 2019, Vancouver, BC, Canada*, pp. 4838–4847, 2019.

Tay, Y., Bahri, D., Yang, L., Metzler, D., and Juan, D. Sparse sinkhorn attention. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pp. 9438–9447. PMLR, 2020.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. Attention is all you need. *CoRR*, abs/1706.03762, 2017.

Wang, S., Li, B. Z., Khabsa, M., Fang, H., and Ma, H. Linformer: Self-attention with linear complexity. *CoRR*, abs/2006.04768, 2020.

Wen, Q., Gao, J., Song, X., Sun, L., Xu, H., and Zhu, S. RobustSTL: A robust seasonal-trend decomposition algorithm for long time series. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 5409–5416, 2019.

Wen, Q., He, K., Sun, L., Zhang, Y., Ke, M., and Xu, H. RobustPeriod: Time-frequency mining for robust multiple periodicity detection. In *Proceedings of the 2021 International Conference on Management of Data (SIGMOD '21)*, pp. 205–215, 2021.

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. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, pp. 101–112, 2021.

Xiong, Y., Zeng, Z., Chakraborty, R., Tan, M., Fung, G., Li, Y., and Singh, V. Nyströmformer: A nyström-based algorithm for approximating self-attention. In *Thirty-Fifth AAAI Conference on Artificial Intelligence*, pp. 14138–14148, 2021.

Zaheer, M., Guruganesh, G., Dubey, K. A., Ainslie, J., Alberti, C., Ontañón, S., Pham, P., Ravula, A., Wang, Q., Yang, L., and Ahmed, A. Big bird: Transformers for

longer sequences. In *Annual Conference on Neural Information Processing Systems (NeurIPS), December 6-12, 2020, virtual*, 2020.

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 *The Thirty-Fifth AAAI Conference on Artificial Intelligence (AAAI)*, volume 35, pp. 11106–11115, 2021.

Zhu, Z. and Soricut, R. H-transformer-1d: Fast one-dimensional hierarchical attention for sequences. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics (ACL) 2021, Virtual Event, August 1-6, 2021*, pp. 3801–3815, 2021.

A. Low-rank Approximation of Attention

In this section, we discuss the low-rank approximation of the attention mechanism. First, we present the Restricted Isometry Property (RIP) matrices whose approximate error bound could be theoretically given in A.1. Then in A.2, we follow prior work and present how to leverage RIP matrices and attention mechanisms.

If the signal of interest is sparse or compressible on a fixed basis, then it is possible to recover the signal from fewer measurements. (Wang et al., 2020; Xiong et al., 2021) suggest that the attention matrix is low-rank, so the attention matrix can be well approximated if being projected into a subspace where the attention matrix is sparse. For the efficient computation of the attention matrix, how to properly select the basis of the projection yet remains to be an open question. The basis which follows the RIP is a potential candidate.

A.1. RIP Matrices

The definition of the RIP matrices is:

Definition A.1. *RIP matrices. Let $m < n$ be positive integers, Φ be a $m \times n$ matrix with real entries, $\delta > 0$, and $K < m$ be an integer. We say that Φ is $(K, \delta) - RIP$, if for every K-sparse vector $x \in \mathbb{R}^n$ we have $(1-\delta)\|x\| \leq \|\Phi x\| \leq (1+\delta)\|x\|$.*

RIP matrices are the matrices that satisfy the restricted isometry property, discovered by D. Donoho, E. Candès and T. Tao in the field of compressed sensing. RIP matrices might be good choices for low-rank approximation because of their good properties. A random matrix has a negligible probability of not satisfying the RIP and many kinds of matrices have proven to be RIP, for example, Gaussian basis, Bernoulli basis, and Fourier basis.

Theorem 2. *Let $m < n$ be positive integers, $\delta > 0$, and $K = O(\frac{m}{log^4 n})$. Let Φ be the random matrix defined by one of the following methods:*

(Gaussian basis) Let the entries of Φ be i.i.d. with a normal distribution $N(0, \frac{1}{m})$.

(Bernoulli basis) Let the entries of Φ be i.i.d. with a Bernoulli distribution taking the values $\pm\frac{1}{\sqrt{m}}$ m, each with 50% probability.

(Random selected Discrete Fourier basis) Let $A \subset \{0, ..., n-1\}$ be a random subset of size m. Let Φ be the matrix obtained from the Discrete Fourier transform matrix (i.e. the matrix F with entries $F[l, j] = \exp^{-2\pi i l j/n}/\sqrt{n}$) for $l, j \in \{0, .., n-1\}$ by selecting the rows indexed by A.

Then Φ is $(K, \sigma) - RIP$ with probability $p \approx 1 - e^{-n}$.

Theorem 2 states that Gaussian basis, Bernoulli basis and

Fourier basis follow RIP. In the following section, the Fourier basis is used as an example and show how to use RIP basis in low-rank approximation in the attention mechanism.

A.2. Low-rank Approximation with Fourier Basis/Legendre Polynomials

Linformer (Wang et al., 2020) demonstrates that the attention mechanism can be approximated by a low-rank matrix. Linformer uses a trainable kernel initialized with Gaussian distribution for the low-rank approximation, While our proposed FEDformer uses Fourier basis/Legendre Polynomials, Gaussian basis, Fourier basis, and Legendre Polynomials all obey RIP, so similar conclusions could be drawn.

Starting from Johnson-Lindenstrauss lemma (Johnson, 1984) and using the version from (Arriaga & Vempala, 2006), Linformer proves that a low-rank approximation of the attention matrix could be made.

Let $\Phi \in \mathbb{R}^{N\times M}$ be the random selected Fourier basis/Legendre Polynomials. Φ is RIP matrix. Referring to Theorem 2, with a probability $p \approx 1-e^{-n}$, for any $x \in \mathbb{R}^N$, we have

$$(1-\delta)\|x\| \leq \|\Phi x\| \leq (1+\delta)\|x\|. \tag{9}$$

Referring to (Arriaga & Vempala, 2006), with a probability $p \approx 1-4e^{-n}$, for any $x_1, x_2 \in \mathbb{R}^N$, we have

$$(1-\delta)\|x_1 x_2^\top\| \leq \|x_1\Phi^\top \Phi x_2^\top\| \leq (1+\delta)\|x_1 x_2^\top\|. \tag{10}$$

With the above inequation function, we now discuss the case in attention mechanism. Let the attention matrix $B = softmax(\frac{QK^\top}{\sqrt{d}}) = exp(A)\cdot D_A^{-1}$, where $(D_A)_{ii} = \sum_{n=1}^{N} exp(A_{ni})$. Following Linformer, we can conclude a theorem as (please refer to (Wang et al., 2020) for the detailed proof)

Theorem 3. *For any row vector $p \in \mathbb{R}^N$ of matrix B and any column vector $v \in \mathbb{R}^N$ of matrix V, with a probability $p = 1-o(1)$, we have*

$$\|b\Phi^\top \Phi v^\top - bv^\top\| \leq \delta\|bv^\top\|. \tag{11}$$

Theorem 3 points out the fact that, using Fourier basis/Legendre Polynomials Φ between the multiplication of attention matrix (P) and values (V), the computation complexity can be reduced from $O(N^2 d)$ to $O(NMd)$, where d is the hidden dimension of the matrix. In the meantime, the error of the low-rank approximation is bounded. However, Theorem 3 only discussed the case which is without the activation function.

Furthermore, with the Cauchy inequality and the fact that the exponential function is Lipchitz continuous in a compact region (please refer to (Wang et al., 2020) for the proof), we can draw the following theorem:

Theorem 4. *For any row vector $A_i \in \mathbb{R}^N$ in matrix A ($A = \frac{QK^\top}{\sqrt{d}}$), with a probability of $p = 1-o(1)$, we have*

$$\|exp(A_i\Phi^\top)\Phi v^\top - exp(A_i)v^\top\| \leq \delta\|exp(A_i)v^\top\|. \tag{12}$$

Theorem 4 states that with the activation function (softmax), the above discussed bound still holds.

In summary, we can leverage RIP matrices for low-rank approximation of attention. Moreover, there exists theoretical error bound when using a randomly selected Fourier basis for low-rank approximation in the attention mechanism.

B. Fourier Component Selection

Let $X_1(t), \ldots, X_m(t)$ be m time series. By applying Fourier transform to each time series, we turn each $X_i(t)$ into a vector $a_i = (a_{i,1}, \ldots, a_{i,d})^\top \in \mathbb{R}^d$. By putting all the Fourier transform vectors into a matrix, we have $A = (a_1, a_2, \ldots, a_m)^\top \in \mathbb{R}^{m\times d}$, with each row corresponding to a different time series and each column corresponding to a different Fourier component. Here, we propose to select s components from the d Fourier components ($s < d$) uniformly at random. More specifically, we denote by $i_1 < i_2 < \ldots < i_s$ the randomly selected components. We construct matrix $S \in \{0,1\}^{s\times d}$, with $S_{i,k} = 1$ if $i = i_k$ and $S_{i,k} = 0$ otherwise. Then, our representation of multivariate time series becomes $A' = AS^\top \in \mathbb{R}^{m\times s}$. The following theorem shows that, although the Fourier basis is randomly selected, under a mild condition, A' can preserve most of the information from A.

Theorem 5. *Assume that $\mu(A)$, the coherence measure of matrix A, is $\Omega(k/n)$. Then, with a high probability, we have*

$$|A - P_{A'}(A)| \leq (1+\epsilon)|A - A_k|$$

if $s = O(k^2/\epsilon^2)$.

Proof. Following the analysis in Theorem 3 from (Drineas et al., 2007), we have

$$\begin{aligned} |A - P_{A'}(A)| &\leq |A - A'(A')^\dagger A_k| \\ &= |A - (AS^\top)(AS^\top)^\dagger A_k| \\ &= |A - (AS^\top)(A_k S^\top)^\dagger A_k|. \end{aligned}$$

Using Theorem 5 from (Drineas et al., 2007), we have, with a probability at least 0.7,

$$|A - (AS^\top)(A_k S^\top)^\dagger A_k| \leq (1+\epsilon)|A - A_k|$$

if $s = O(k^2/\epsilon^2 \times \mu(A)n/k)$. The theorem follows because $\mu(A) = O(k/n)$. □

C. Wavelets

In this section, we present some technical background about Wavelet transform which is used in our proposed framework.

C.1. Continuous Wavelet Transform

First, let's see how a function $f(t)$ is decomposed into a set of basis functions $\psi_{\mathrm{s},\tau}(t)$, called the wavelets. It is known as the continuous wavelet transform or CWT. More formally it is written as

$$\gamma(s,\tau) = \int f(t)\Psi^*_{s,\tau}(t)dt,$$

where * denotes complex conjugation. This equation shows the variables $\gamma(s,\tau)$, s and τ are the new dimensions, scale, and translation after the wavelet transform, respectively.

The wavelets are generated from a single basic wavelet $\Psi(t)$, the so-called mother wavelet, by scaling and translation as

$$\psi_{s,\tau}(t) = \frac{1}{\sqrt{s}}\psi\left(\frac{t-\tau}{s}\right),$$

where s is the scale factor, τ is the translation factor, and $\sqrt{s}$ is used for energy normalization across the different scales.

C.2. Discrete Wavelet Transform

Continues wavelet transform maps a one-dimensional signal to a two-dimensional time-scale joint representation which is highly redundant. To overcome this problem, people introduce discrete wavelet transformation (DWT) with mother wavelet as

$$\psi_{j,k}(t) = \frac{1}{\sqrt{s_0^j}}\psi\left(\frac{t-k\tau_0 s_0^j}{s_0^j}\right)$$

DWT is not continuously scalable and translatable but can be scaled and translated in discrete steps. Here j and k are integers and $s_0 > 1$ is a fixed dilation step. The translation factor τ_0 depends on the dilation step. The effect of discretizing the wavelet is that the time-scale space is now sampled at discrete intervals. We usually choose $s_0 = 2$ so that the sampling of the frequency axis corresponds to dyadic sampling. For the translation factor, we usually choose $\tau_0 = 1$ so that we also have a dyadic sampling of the time axis.

When discrete wavelets are used to transform a continuous signal, the result will be a series of wavelet coefficients and it is referred to as the wavelet decomposition.

C.3. Orthogonal Polynomials

The next thing we need to focus on is orthogonal polynomials (OPs), which will serve as the mother wavelet function we introduce before. A lot of properties have to be maintained to be a mother wavelet, like admissibility condition, regularity conditions, and vanishing moments. In short, we are interested in the OPs that are non-zero over a finite domain and are zero almost everywhere else. Legendre is a popular set of OPs used it in our work here. Some other popular OPs can also be used here like Chebyshev without much modification.

C.4. Legendre Polynomails

The Legendre polynomials are defined with respect to (w.r.t.) a uniform weight function $w_L(x) = 1$ for $-1 \leqslant x \leqslant 1$ or $w_L(x) = \mathbf{1}_{[-1,1]}(x)$ such that

$$\int_{-1}^{1} P_i(x)P_j(x)dx = \begin{cases} \frac{2}{2i+1} & i = j, \\ 0 & i \neq j. \end{cases}$$

Here the function is defined over $[-1,1]$, but it can be extended to any interval $[a,b]$ by performing different shift and scale operations.

C.5. Multiwavelets

The multiwavelets which we use in this work combine advantages of the wavelet and OPs we introduce before. Other than projecting a given function onto a single wavelet function, multiwavelet projects it onto a subspace of degree-restricted polynomials. In this work, we restricted our exploration to one family of OPs: Legendre Polynomials.

First, the basis is defined as: A set of orthonormal basis w.r.t. measure μ, are $\phi_0, \ldots, \phi_{k-1}$ such that $\langle \phi_i, \phi_j \rangle_\mu = \delta_{ij}$. With a specific measure (weighting function $w(x)$), the orthonormality condition can be written as $\int \phi_i(x)\phi_j(x)w(x)dx = \delta_{ij}$.

Follow the derivation in (Gupta et al., 2021), through using the tools of Gaussian Quadrature and Gram-Schmidt Orthogonalizaition, the filter coefficients of multiwavelets using Legendre polynomials can be written as

$$\begin{aligned} H_{ij}^{(0)} &= \sqrt{2}\int_0^{1/2} \phi_i(x)\phi_j(2x)w_L(2x-1)dx \\ &= \frac{1}{\sqrt{2}}\int_0^1 \phi_i(x/2)\phi_j(x)dx \\ &= \frac{1}{\sqrt{2}}\sum_{i=1}^{k} \omega_i \phi_i\left(\frac{x_i}{2}\right)\phi_j\left(x_i\right). \end{aligned}$$

For example, if $k = 3$, following the formula, the filter

coefficients are derived as follows

$$H^0 = \begin{bmatrix} \frac{1}{\sqrt{2}} & 0 & 0 \\ -\frac{\sqrt{3}}{2\sqrt{2}} & \frac{1}{2\sqrt{2}} & 0 \\ 0 & -\frac{\sqrt{15}}{4\sqrt{2}} & \frac{1}{4\sqrt{2}} \end{bmatrix}, H^1 = \begin{bmatrix} \frac{1}{\sqrt{2}} & 0 & 0 \\ \frac{\sqrt{3}}{2\sqrt{2}} & \frac{1}{2\sqrt{2}} & 0 \\ 0 & \frac{\sqrt{15}}{4\sqrt{2}} & \frac{1}{4\sqrt{2}} \end{bmatrix},$$

$$G^0 = \begin{bmatrix} \frac{1}{2\sqrt{2}} & \frac{\sqrt{3}}{2\sqrt{2}} & 0 \\ 0 & \frac{1}{4\sqrt{2}} & \frac{\sqrt{15}}{4\sqrt{2}} \\ 0 & 0 & \frac{1}{\sqrt{2}} \end{bmatrix}, G^1 = \begin{bmatrix} -\frac{1}{2\sqrt{2}} & \frac{\sqrt{3}}{2\sqrt{2}} & 0 \\ 0 & -\frac{1}{4\sqrt{2}} & \frac{\sqrt{15}}{4\sqrt{2}} \\ 0 & 0 & -\frac{1}{\sqrt{2}} \end{bmatrix}$$

For a given $f(x)$, the multiwavelet coefficients at the scale n can be defined as $\mathbf{s}_l^n = \left[\langle f, \phi_{il}^n \rangle_{\mu_n}\right]_{i=0}^{k-1}$, $\mathbf{d}_l^n = \left[\langle f, \psi_{il}^n \rangle_{\mu_n}\right]_{i=0}^{k-1}$, respectively, w.r.t. measure μ_n with $\mathbf{s}_l^n, \mathbf{d}_l^n \in \mathbb{R}^{k \times 2^n}$. ϕ_{il}^n are wavelet orthonormal basis of piecewise polynomials. The decomposition/reconstruction across scales is defined as

$$\begin{aligned} \mathbf{s}_l^n &= H^{(0)} \mathbf{s}_{2l}^{n+1} + H^{(1)} \mathbf{s}_{2l+1}^{n+1}, \\ \mathbf{s}_{2l}^{n+1} &= \Sigma^{(0)} \left(H^{(0)T} \mathbf{s}_l^n + G^{(0)T} \mathbf{d}_l^n \right), \\ \mathbf{d}_l^n &= G^{(0)} \mathbf{s}_{2l}^{n+1} + H^{(1)} \mathbf{s}_{2l+1}^{n+1}, \\ \mathbf{s}_{2l+1}^{n+1} &= \Sigma^{(1)} \left(H^{(1)T} \mathbf{s}_l^n + G^{(1)T} \mathbf{d}_l^n \right), \end{aligned} \tag{13}$$

where $\left(H^{(0)}, H^{(1)}, G^{(0)}, G^{(1)}\right)$ are linear coefficients for multiwavelet decomposition filters. They are fixed matrices used for wavelet decomposition. The multiwavelet representation of a signal can be obtained by the tensor product of multiscale and multiwavelet basis. Note that the basis at various scales are coupled by the tensor product, so we need to untangle it. Inspired by Gupta et al.,2021, we adapt a non-standard wavelet representation to reduce the model complexity. For a map function $F(x) = x'$, the map under multiwavelet domain can be written as

$$U_{dl}^n = A_n d_l^n + B_n s_l^n, \quad U_{sl}^n = C_n d_l^n, \quad U_{sl}^L = \bar{F} s_l^L, \tag{14}$$

where $(U_{sl}^n, U_{dl}^n, s_l^n, d_l^n)$ are the multiscale, multiwavelet coefficients, L is the coarsest scale under recursive decomposition, and A_n, B_n, C_n are three independent FEB-f blocks modules used for processing different signal during decomposition and reconstruction. Here $\bar{F}$ is a single-layer of perceptrons which processes the remaining coarsest signal after L decomposed steps.

C.6. Frequency Enhanced Attention with Wavelet Transform (FEA-w)

For each cycle L, it produces a processed high-frequency tensor $Ud(L)$, a processed low-frequency frequency tensor $Us(L)$, and the raw low-frequency tensor $X(L+1)$. This is a ladder-down approach, and the decomposition stage performs the decimation of the signal by a factor of 1/2, running for a maximum of L cycles, where $L < \log_2(M)$ for a given input sequence of size M. In practice, L is set as a fixed argument parameter. The three sets of FEB-f blocks are shared during different decomposition cycles L. For the wavelet reconstruction part, we recursively build up our output tensor as well. For each cycle L, we combine $X(L+1)$, $Us(L)$, and $Ud(L)$ produced from the decomposition part and produce $X(L)$ for the next reconstruction cycle. For each cycle, the length dimension of the signal tensor is increased by 2 times.

D. Output Distribution Analysis

D.1. Bad Case Analysis

Using vanilla Transformer as baseline model, we demonstrate two bad long-term series forecasting cases in ETTm1 dataset as shown in the following Figure 7.

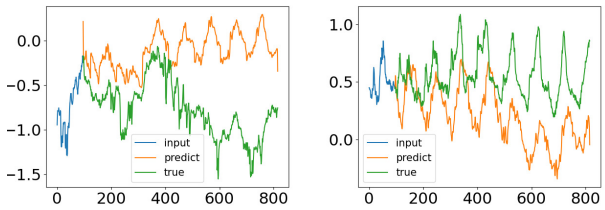

Figure 7. Different distribution between ground truth and forecasting output from vanilla Transformer in a real-world ETTm1 dataset. Left: frequency mode and trend shift. Right: trend shift.

The forecasting shifts in Figure 7 is particularly related to the point-wise generation mechanism adapted by the vanilla Transformer model. To the contrary of classic models like Autoregressive integrated moving average (ARIMA) which has a predefined data bias structure for output distribution, Transformer-based models forecast each point independently and solely based on the overall MSE loss learning. This would result in different distribution between ground truth and forecasting output in some cases, leading to performance degradation.

D.2. Kolmogorov-Smirnov Test

We adopt Kolmogorov-Smirnov (KS) test to check whether the two data samples come from the same distribution. KS test is a nonparametric test of the equality of continuous or discontinuous, two-dimensional probability distributions. In essence, the test answers the question "what is the probability that these two sets of samples were drawn from the same (but unknown) probability distribution". It quantifies a distance between the empirical distribution function of two samples. The Kolmogorov-Smirnov statistic is

$$D_{n,m} = \sup_x |F_{1,n}(x) - F_{2,m}(x)|$$

where $F_{1,n}$ and $F_{2,m}$ are the empirical distribution functions of the first and the second sample respectively, and sup is the

Table 6. Kolmogrov-Smirnov test P value for long sequence time-series forecasting output on ETT dataset (full experiment)

Methods		Transformer	LogTrans	Informer	Reformer	Autoformer	FEDformer	True
ETTm1	96	0.0090	0.0073	0.0055	0.0055	0.020	**0.048**	0.023
	192	0.0052	0.0043	0.0029	0.0013	0.015	**0.028**	0.013
	336	0.0022	0.0026	0.0019	0.0006	0.012	**0.015**	0.010
	720	0.0023	0.0064	0.0016	0.0011	0.008	**0.014**	0.004
ETTm2	96	0.0012	0.0025	0.0008	0.0028	**0.078**	0.071	0.087
	192	0.0011	0.0011	0.0006	0.0015	**0.047**	0.045	0.060
	336	0.0005	0.0011	0.00009	0.0007	0.027	**0.028**	0.042
	720	0.0008	0.0005	0.0002	0.0005	**0.023**	0.021	0.023
Count		0	0	0	0	3	5	NA

supremum function. For large samples, the null hypothesis is rejected at level α if

$$D_{n,m} > \sqrt{-\frac{1}{2}\ln\left(\frac{\alpha}{2}\right)} \cdot \sqrt{\frac{n+m}{n \cdot m}},$$

where n and m are the sizes of the first and second samples respectively.

D.3. Distribution Experiments and Analysis

Though the KS test omits the temporal information from the input and output sequence, it can be used as a tool to measure the global property of the foretasting output sequence compared to the input sequence. The null hypothesis is that the two samples come from the same distribution. We can tell that if the P-value of the KS test is large and then the null hypothesis is less likely to be rejected for true output distribution.

We applied KS test on the output sequence of 96-720 prediction tasks for various models on the ETTm1 and ETTm2 datasets, and the results are summarized in Table 6. In the test, we compare the fixed 96-time step input sequence distribution with the output sequence distribution of different lengths. Using a 0.01 P-value as statistics, various existing Transformer baseline models have much less P-value than 0.01 except Autoformer, which indicates they have a higher probability to be sampled from the different distributions. Autoformer and FEDformer have much larger P value compared to other models, which mainly contributes to their seasonal trend decomposition mechanism. Though we get close results from ETTm1 by both models, the proposed FEDformer has much larger P-values in ETTm1. And it is the only model whose null hypothesis can not be rejected with P-value larger than 0.01 in all cases of the two datasets, implying that the output sequence generated by FEDformer shares a more similar distribution as the input sequence than others and thus justifies the our design motivation of FEDformer as discussed in Section 1.

Note that in the ETTm1 dataset, the True output sequence has a smaller P-value compared to our FEDformer's predicted output, it shows that the model's close output distribution is achieved through model's control other than merely more accurate prediction. This analysis shed some light on why the seasonal-trend decomposition architecture can give us better performance in long-term forecasting. The design is used to constrain the trend (mean) of the output distribution. Inspired by such observation, we design frequency enhanced block to constrain the seasonality (frequency mode) of the output distribution.

Table 7. Summarized feature details of six datasets.

DATASET	LEN	DIM	FREQ
ETTM2	69680	8	15 MIN
ELECTRICITY	26304	322	1H
EXCHANGE	7588	9	1 DAY
TRAFFIC	17544	863	1H
WEATHER	52696	22	10 MIN
ILI	966	8	7 DAYS

E. Supplemental Experiments

E.1. Dataset Details

In this paragraph, the details of the experiment datasets are summarized as follows: 1) ETT (Zhou et al., 2021) dataset contains two sub-dataset: ETT1 and ETT2, collected from two electricity transformers at two stations. Each of them has two versions in different resolutions (15min & 1h). ETT dataset contains multiple series of loads and one series of oil temperatures. 2) Electricity[1] dataset contains the electricity consumption of clients with each column corresponding to one client. 3) Exchange (Lai et al., 2018) contains the current exchange of 8 countries. 4) Traffic[2] dataset contains the occupation rate of freeway system across the State of

[1]https://archive.ics.uci.edu/ml/datasets/ElectricityLoadDiagrams20112014

[2]http://pems.dot.ca.gov

Table 8. Multivariate long sequence time-series forecasting results on ETT full benchmark. The best results are highlighted in bold.

Methods		FEDformer-f		FEDformer-w		Autoformer		Informer		LogTrans		Reformer	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
$ETTh1$	96	**0.376**	**0.419**	0.395	0.424	0.449	0.459	0.865	0.713	0.878	0.740	0.837	0.728
	192	**0.420**	**0.448**	0.469	0.470	0.500	0.482	1.008	0.792	1.037	0.824	0.923	0.766
	336	**0.459**	**0.465**	0.530	0.499	0.521	0.496	1.107	0.809	1.238	0.932	1.097	0.835
	720	**0.506**	**0.507**	0.598	0.544	0.514	0.512	1.181	0.865	1.135	0.852	1.257	0.889
$ETTh2$	96	**0.346**	**0.388**	0.394	0.414	0.358	0.397	3.755	1.525	2.116	1.197	2.626	1.317
	192	**0.429**	**0.439**	0.439	0.445	0.456	0.452	5.602	1.931	4.315	1.635	11.12	2.979
	336	0.496	0.487	**0.482**	**0.480**	0.482	0.486	4.721	1.835	1.124	1.604	9.323	2.769
	720	**0.463**	**0474**	0.500	0.509	0.515	0.511	3.647	1.625	3.188	1.540	3.874	1.697
$ETTm1$	96	0.379	0.419	**0.378**	**0.418**	0.505	0.475	0.672	0.571	0.600	0.546	0.538	0.528
	192	**0.426**	**0.441**	0.464	0.463	0.553	0.496	0.795	0.669	0.837	0.700	0.658	0.592
	336	**0.445**	**0.459**	0.508	0.487	0.621	0.537	1.212	0.871	1.124	0.832	0.898	0.721
	720	**0.543**	**0.490**	0.561	0.515	0.671	0.561	1.166	0.823	1.153	0.820	1.102	0.841
$ETTm2$	96	**0.203**	**0.287**	0.204	0.288	0.255	0.339	0.365	0.453	0.768	0.642	0.658	0.619
	192	**0.269**	**0.328**	0.316	0.363	0.281	0.340	0.533	0.563	0.989	0.757	1.078	0.827
	336	**0.325**	**0.366**	0.359	0.387	0.339	0.372	1.363	0.887	1.334	0.872	1.549	0.972
	720	**0.421**	**0.415**	0.433	0.432	0.422	0.419	3.379	1.338	3.048	1.328	2.631	1.242

California. 5) Weather[3] dataset contains 21 meteorological indicators for a range of 1 year in Germany. 6) Illness[4] dataset contains the influenza-like illness patients in the United States. Table 7 summarizes feature details (Sequence Length: Len, Dimension: Dim, Frequency: Freq) of the six datasets. All datasets are split into the training set, validation set and test set by the ratio of 7:1:2.

E.2. Implementation Details

Our model is trained using ADAM (Kingma & Ba, 2017) optimizer with a learning rate of $1e^{-4}$. The batch size is set to 32. An early stopping counter is employed to stop the training process after three epochs if no loss degradation on the valid set is observed. The mean square error (MSE) and mean absolute error (MAE) are used as metrics. All experiments are repeated 5 times and the mean of the metrics is used in the final results. All the deep learning networks are implemented in PyTorch (Paszke et al., 2019) and trained on NVIDIA V100 32GB GPUs.

E.3. ETT Full Benchmark

We present the full-benchmark on the four ETT datasets (Zhou et al., 2021) in Table 8 (multivariate forecasting) and Table 9 (univariate forecasting). The ETTh1 and ETTh2 are recorded hourly while ETTm1 and ETTm2 are recorded every 15 minutes. The time series in ETTh1 and ETTm1 follow the same pattern, and the only difference is the sampling rate, similarly for ETTh2 and ETTm2. On average, our FEDformer yields a **11.5%** relative MSE reduction for multivariate forecasting, and a **9.4%** reduction for univariate forecasting over the SOTA results from Autoformer.

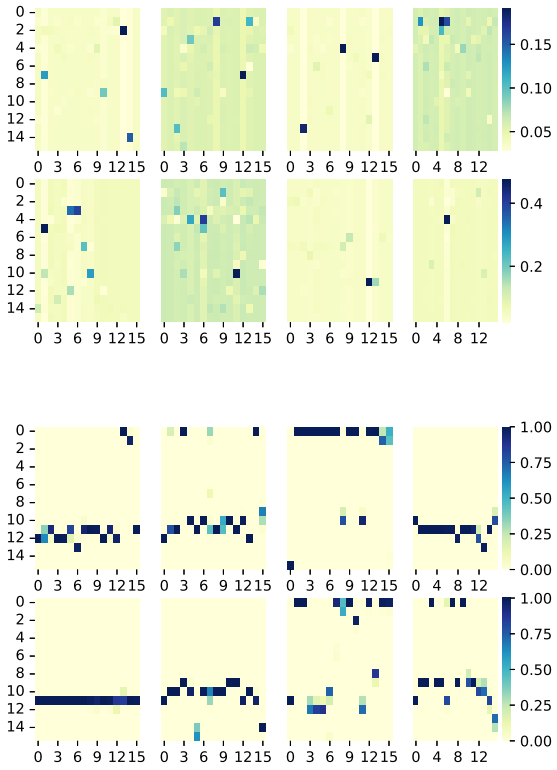

Figure 8. Multihead attention map with 8 heads using tanh (top) and softmax (bottom) as activation map for the FEDformer-f training on ETTm2 dataset.

[3]https://www.bgc-jena.mpg.de/wetter/

[4]https://gis.cdc.gov/grasp/fluview/fluportaldashboard.html

Table 9. Univariate long sequence time-series forecasting results on ETT full benchmark. The best results are highlighted in bold.

Methods		FEDformer-f		FEDformer-w		Autoformer		Informer		LogTrans		Reformer	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
$ETTh1$	96	0.079	0.215	0.080	0.214	**0.071**	**0.206**	0.193	0.377	0.283	0.468	0.532	0.569
	192	**0.104**	**0.245**	0.105	0.256	0.114	0.262	0.217	0.395	0.234	0.409	0.568	0.575
	336	0.119	0.270	0.120	0.269	**0.107**	**0.258**	0.202	0.381	0.386	0.546	0.635	0.589
	720	0.142	0.299	0.127	0.280	**0.126**	**0.283**	0.183	0.355	0.475	0.628	0.762	0.666
$ETTh2$	96	**0.128**	**0.271**	0.156	0.306	0.153	0.306	0.213	0.373	0.217	0.379	1.411	0.838
	192	**0.185**	**0.330**	0.238	0.380	0.204	0.351	0.227	0.387	0.281	0.429	5.658	1.671
	336	**0.231**	**0.378**	0.271	0.412	0.246	0.389	0.242	0.401	0.293	0.437	4.777	1.582
	720	0.278	0.420	0.288	0.438	**0.268**	**0.409**	0.291	0.439	0.218	0.387	2.042	1.039
$ETTm1$	96	**0.033**	**0.140**	0.036	0.149	0.056	0.183	0.109	0.277	0.049	0.171	0.296	0.355
	192	**0.058**	**0.186**	0.069	0.206	0.081	0.216	0.151	0.310	0.157	0.317	0.429	0.474
	336	0.084	0.231	**0.071**	**0.209**	0.076	0.218	0.427	0.591	0.289	0.459	0.585	0.583
	720	**0.102**	0.250	0.105	**0.248**	0.110	0.267	0.438	0.586	0.430	0.579	0.782	0.730
$ETTm2$	96	0.067	0.198	**0.063**	**0.189**	0.065	0.189	0.088	0.225	0.075	0.208	0.076	0.214
	192	**0.102**	**0.245**	0.110	0.252	0.118	0.256	0.132	0.283	0.129	0.275	0.132	0.290
	336	**0.130**	**0.279**	0.147	0.301	0.154	0.305	0.180	0.336	0.154	0.302	0.160	0.312
	720	**0.178**	**0.325**	0.219	0.368	0.182	0.335	0.300	0.435	0.160	0.321	0.168	0.335

E.4. Cross Attention Visualization

The $\sigma(\tilde{\boldsymbol{Q}} \cdot \tilde{\boldsymbol{K}}^{\top})$ can be viewed as the cross attention weight for our proposed frequency enhanced cross attention block. Several different activation functions can be used for attention matrix activation. Tanh and softmax are tested in this work with various performances on different datasets. We use tanh as the default one. Different attention patterns are visualized in Figure 8. Here two samples of cross attention maps are shown for FEDformer-f training on the ETTm2 dataset using tanh and softmax respectively. It can be seen that attention with Softmax as activation function seems to be more sparse than using tanh. Overall we can see attention in the frequency domain is much sparser compared to the normal attention graph in the time domain, which indicates our proposed attention can represent the signal more compactly. Also this compact representation supports our random mode selection mechanism to achieve linear complexity.

E.5. Improvements of Mixture of Experts Decomposition

We design a mixture of experts decomposition mechanism which adopts a set of average pooling layers to extract the trend and a set of data-dependent weights to combine them. The default average pooling layers contain filters with kernel size 7, 12, 14, 24 and 48 respectively. For comparison, we use single expert decomposition mechanism which employs a single average pooling layer with a fixed kernel size of 24 as the baseline. In Table 10, a comparison study of multivariate forecasting is shown using FEDformer-f model on two typical datasets. It is observed that the designed mixture of experts decomposition brings better performance than the single decomposition scheme.

Table 10. Performance improvement of the designed mixture of experts decomposition scheme.

Methods	FEDformer-f		FEDformer-f	
Dataset	ETTh1		Weather	
Mechanism	MOE	Single	MOE	Single
96	**0.217**	0.238	0.376	**0.375**
192	**0.276**	0.291	0.420	**0.412**
336	**0.339**	0.352	**0.450**	0.455
720	**0.403**	0.413	**0.496**	0.502
Improvement	5.35%		0.57%	

E.6. Multiple random runs

Table 11 lists both mean and standard deviation (STD) for FEDformer-f and Autoformer with 5 runs. We observe a small variance in the performance of FEDformer-f, despite the randomness in frequency selection.

Table 11. A subset of the benchmark showing both Mean and STD.

MSE		ETTm2	Electricity	Exchange	Traffic
FED-f	96	0.203 ± 0.0042	0.194 ± 0.0008	0.148 ± 0.002	0.217 ± 0.008
	192	0.269 ± 0.0023	0.201± 0.0015	0.270± 0.008	0.604 ± 0.004
	336	0.325 ± 0.0015	0.215± 0.0018	0.460± 0.016	0.621 ± 0.006
	720	0.421 ± 0.0038	0.246± 0.0020	1.195± 0.026	0.626 ± 0.003
Autoformer	96	0.255 ± 0.020	0.201± 0.003	0.197± 0.019	0.613± 0.028
	192	0.281 ± 0.027	0.222± 0.003	0.300± 0.020	0.616± 0.042
	336	0.339 ± 0.018	0.231± 0.006	0.509± 0.041	0.622± 0.016
	720	0.422 ± 0.015	0.254± 0.007	1.447± 0.084	0.419± 0.017

E.7. Sensitivity to the number of modes: ETTx1 vs ETTx2

The choice of modes number depends on data complexity. The time series that exhibits the higher complex patterns

requires the larger the number of modes. To verify this claim, we summarize the complexity of ETT datasets, measured by permutation entropy and SVD entropy, in Table 12. It is observed that ETTx1 has a significantly higher complexity (corresponding to a higher entropy value) than ETTx2, thus requiring a larger number of modes.

Table 12. Complexity experiments for datasets

Methods	ETTh1	ETTh2	ETTm1	ETTm2
Permutation Entropy	0.954	0.866	0.959	0.788
SVD Entropy	0.807	0.495	0.589	0.361

E.8. When Fourier/Wavelet model performs better

Our high level principle of model deployment is that Fourier-based model is usually better for less complex time series, while wavelet is normally more suitable for complex ones. Specifically, we found that wavelet-based model is more effective on multivariate time series, while Fourier-based one normally achieves better results on univariate time series. As indicated in Table 13, complexity measures on multivariate time series are higher than those on univariate ones.

Table 13. Perm Entropy Complexity comparison for multi vs uni

Permutation Entropy	Electricity	Traffic	Exchange	Illness
Multivariate	0.910	0.792	0.961	0.960
Univariate	0.902	0.790	0.949	0.867