Permutation Dependent Feature Mixing for Multivariate Time Series Forecasting

Rikuto Yamazono[1](✉) and Hirotake Hachiya[1,2]

[1] Wakayama University, 930, Sakaedani, Wakayama-city, Wakayama 640-8510, Japan
s256279@wakayama-u.ac.jp

[2] Center for AIP, RIKEN, Nihonbashi 1-chome Mitsui Building, 15th floor, 1-4-1, Nihonbashi, Chuo-ku, Tokyo 103-0027, Japan

Abstract. Multivariate time series forecasting is critical in finance and meteorology, influencing decision-making. Though effective in capturing long-range dependencies in natural language processing, traditional Transformer models face challenges when applied to time series data, including computational inefficiency and the loss of positional encoding effects. Time-Series Mixer (TSMixer) addresses these issues by efficiently blending the temporal and feature dimensions in multivariate time series data, thereby facilitating sequential dependent feature extraction. However, the current feature mixing in TSMixer applies a common multilayer perception across all time steps, leading to time-invariant, non-adaptive feature exchange that does not allow for accurate extraction of historical information. Therefore, we propose incorporating adaptive frequency components and event proximity as additional information vectors into the Feature Mixing component of TSMixer to improve its capacity to interpret complex feature interrelations. Our research validates the effectiveness of these enhancements through experiments with various real-world multivariate time series datasets, including weather and traffic data, emphasizing its potential across different scenarios. Codes are available at https://github.com/rikuter67/FAM-EPAM.

Keywords: Multivariate Time Series · TSMixer · Time Series Forecasting

1 Introduction

Time series forecasting is indispensable across various sectors, shaping crucial decision-making processes. Traditional methods, such as ARIMA [4,5], rely on predefined models to capture trends and cycles of historical series. While they are effective for stationary series, their fixed structure and inability to capture the dynamic dependencies among multiple features would result in poor performance with real-world data.

Deep neural networks with recurrent architectures, i.e., RNNs [12,15], and memory cells, i.e., LSTMs [7,10], enable to capture dynamic and temporal dependencies by encoding significant information into latent vectors extracted from historical series. However, RNNs with a one-step recurrent connection face limitations on long-term dependencies due to gradient vanishing and exploding issues. While LSTMs with gates and cells can capture longer-term dependencies, their effectiveness is still constrained due to the finite capacity of the memory cells and the complexity of their computational processes.

Transformers [11,14,16,18] introduce an attention mechanism, e.g., self-attention, which allows the model to directly encode important information into the sequence vectors themselves based on the relationship, e.g., co-occurrence within the sequence. This mechanism enables models to process sequences in parallel and provides an enhanced capacity to capture complex and long-term dependencies. However, the inherent permutation invariance of the attention mechanisms poses significant challenges for processing time series data despite the success in natural language processing, which employs a heuristic extension, called positional encoding, to add information about the order of time steps.

In light of this, recent studies have suggested the potential of linear models, which are inherently sensitive to the order of the inputs, for sequential data [17]. Among linear models, the TSMixer (Time-Series Mixer) [6] model employs repeated MLPs (Multi-layer perceptions) to mix time and feature information alternately, encoding useful information into sequence vectors from complex multivariate time series data. However, TSMixer's feature mixing approach uses a common MLP across all time steps, leading to time-invariant, non-adaptive feature mixing, hindering the accurate extraction of historical information.

To address this issue, we propose enhancements to the Feature Mixing component of TSMixer. Firstly, we propose a Frequency-Aware Mixer (FAM), which adds an adaptive frequency component of each feature to the feature-mixing, enabling the adjustment of the strength of feature mixing based on the adaptive time cycles. Secondly, we propose an Event Proximity-Aware Mixer (EPAM), which adds the proximity to the principal observation (event) as an additional component of the feature mixing, enabling the strength of the feature mixing to be adjusted based on the relation to the representative events. These enhancements will enable the model to more accurately grasp the complex interrelations among features.

The main contributions of this paper are summarized as follows:

1. We propose to enhance the Feature Mixing component of TSMixer, the state-of-the-art multivariate time-series forecasting method, to allow for time-dependent and adaptive mixing by introducing principal frequency components, called Frequency-Aware Mixer (FAM) and the distance to principal time-step, called Event Proximity-Aware Mixer (EPAM) as additional information vectors.
2. We demonstrate the effectiveness of the proposed method over existing state-of-the-art multivariate time-series forecasting methods through extensive comparative experiments on various real-world datasets.

After this introductory section, the rest of this paper is organized as follows. Section 2 describes the formulation and reviews related works. Section 3 details

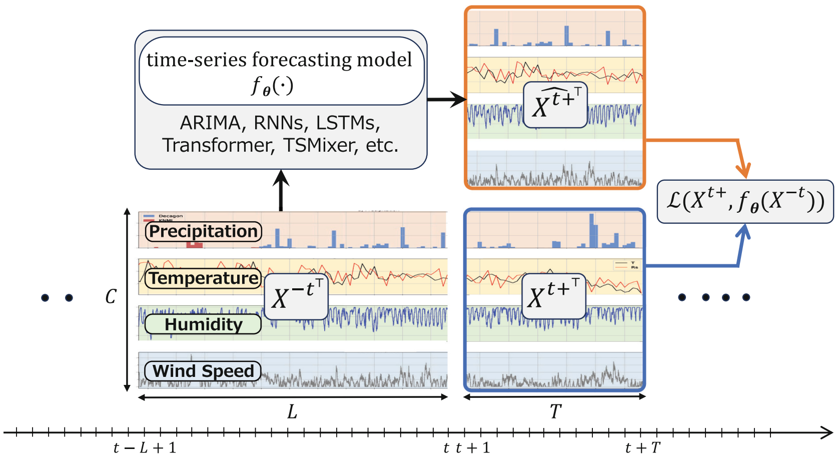


Fig. 1. Illustration of the time series forecasting process based on the formulation using weather-related multivariate data. $f_\theta(\cdot)$ transforms L-step of observations X^{-t} to generate T-step future observations $\widehat{X^{t+}}$, which is then evaluated against the ground truth X^{t+} using the loss function $\mathcal{L}(\cdot, \cdot)$ to compute the discrepancy. Multivariate data are adapted from [8].

the proposed method. Section 4 describes the experimental evaluation and discussion; Sect. 6 presents the conclusion.

2 Formulation and Related Works

This section formulates the problem of multivariate time series forecasting and reviews its related works.

2.1 Formulation

Let X_{tc} denote the c-th observation at time t and : denote all elements at the corresponding axis. Let $X^{-t} \in \mathbb{R}^{L\times C}$ be L-step history of observations where $X^{-t}_{t:} \in \mathbb{R}^{1\times C}$ be a vector of C observations at time step t and $X^{-t}_{:c} \in \mathbb{R}^{L\times 1}$ be a vector of the c-th observation over L-step. Meanwhile, let $X^{t+} \in \mathbb{R}^{T\times C}$ be T-step future observations starting from the next step of X^{-t} as follows:

$$\begin{aligned} X^{-t} &= \left[X_{t-L+1:}, \ldots, X_{t-1:}, X_{t:}\right], \\ X^{t+} &= \left[X_{t+1:}, X_{t+2:} \ldots, X_{t+T:}\right]. \end{aligned} \tag{1}$$

The task of multivariate time-series forecasting is to obtain a model $f(\cdot)$ to predict future observations X^{t+} given its history X^{-t} as follows:

$$\widehat{X^{t+}} = f_\theta(X^{-t}), \tag{2}$$

Parameter $\boldsymbol{\theta}$ of the model is tuned to minimize the loss function $\mathcal{L}(\cdot)$ averaged over training data $\mathcal{D}_{\mathrm{tr}}$ as follows:

$$\min_{\boldsymbol{\theta}} \frac{1}{|\mathcal{D}_{\mathrm{tr}}|} \sum_{t=L}^{|\mathcal{D}_{\mathrm{tr}}|} \mathcal{L}\big(X^{t+}, f_{\boldsymbol{\theta}}(X^{-t})\big), \tag{3}$$

where $\mathcal{D}_{\mathrm{tr}}$ is defined as follows:

$$\mathcal{D}_{\mathrm{tr}} \equiv \Big\{\big(X^{-t}, X^{t+}\big)\Big\}_{t=L}^{N_{\mathrm{tr}}-T}, \tag{4}$$

where N_{tr} is the number of steps in the training sequence. Similarly, the validation and test data are defined as follows:

$$\begin{aligned} \mathcal{D}_{\mathrm{val}} &\equiv \Big\{\big(X^{-t}, X^{t+}\big)\Big\}_{t=N_{\mathrm{tr}}}^{N_{\mathrm{tr}}+N_{\mathrm{val}}-T}, \\ \mathcal{D}_{\mathrm{te}} &\equiv \Big\{\big(X^{-t}, X^{t+}\big)\Big\}_{t=N_{\mathrm{tr}}+N_{\mathrm{val}}}^{N_{\mathrm{tr}}+N_{\mathrm{val}}+N_{\mathrm{te}}-T}, \end{aligned} \tag{5}$$

where N_{val} and N_{te} are the numbers of steps in the validation and test sequence, respectively—there is no overlap between training, validation, and test sequences. As Fig. 1 illustrates the formulation overview.

2.2 Attention Mechanisms in Time Series Forecasting

Attention mechanisms have been applied to time series forecasting [11,14,16,18], enabling models to dynamically encode important information into the sequence vectors X^{-t} based on the relationships between observation vectors at different time steps. More specifically, in the attention mechanism, affinity weight $W^{\mathrm{att}} \in \mathbb{R}^{L\times L}$ is computed based on the similarity between query vectors $Q \in \mathbb{R}^{L\times C}$ and key vectors $K \in \mathbb{R}^{L\times C}$. Next, query Q is transformed through an interpolation of value vectors $V \in \mathbb{R}^{L\times C}$, as follows:

$$\begin{aligned} W^{\mathrm{att}} &= \mathrm{softmax}\Big(\frac{(QW^Q)\,(KW^K)^\top}{\sqrt{C}}\Big), \\ Q' &= \mathrm{Attention}(Q, K, V) = W^{\mathrm{att}}(V\,W^V), \end{aligned} \tag{6}$$

where W^Q, W^K, and $W^V \in \mathbb{R}^{C\times C}$ are trainable linear projection matrices. This weight W^{att} captures the dependencies across the time series, making attention mechanisms particularly useful for identifying intricate temporal relationships.

There are several extensions to overcome the limitation of Transformer for multivariate time-series forecasting, such as Autoformer [16], Informer [18], and PatchPST [11]. Autoformer incorporates an autocorrelation mechanism to capture long-term dependency and a deep decomposition architecture that sequentially decomposes the time series data into trend, seasonal, and random components during forecasting. Informer introduces ProbSparse self-attention, which probabilistically selects important queries with higher attention scores to reduce

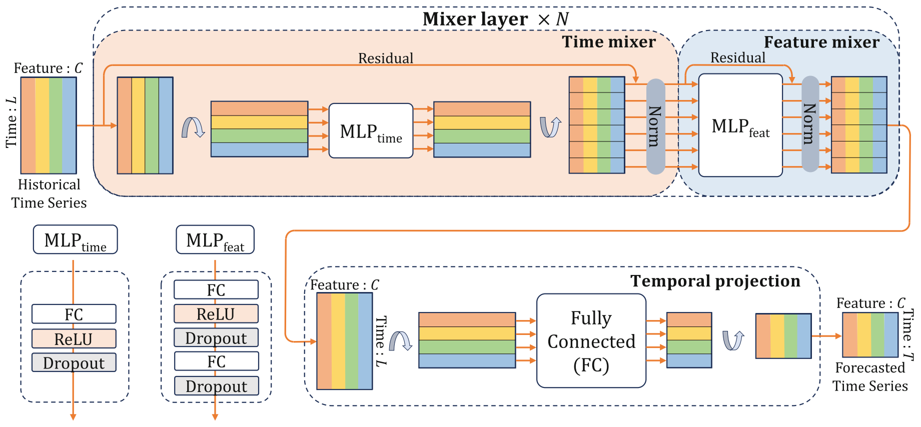


Fig. 2. Architecture of TSMixer, consisting of N repeated mixer layers and a temporal projection. Note that time and feature-mixing MLPs in each mixer layer are shared across all features and all time steps.

computational complexity enabling the efficient capture of long-term dependency. PatchTST divides the time series into patch chunks treated as time steps in the attention mechanism and applies single embedding and attention mechanism individually to each multivariate series, enabling efficient multivariate time-series forecasting.

However, due to the attention mechanisms' inherent permutation invariance property, Transformer models face challenges when directly applied to time series data, where the order of time steps critically impacts forecasting accuracy. Currently, Transformer models attempt to address this issue through positional encoding, which aims to inject sequence information into the model. Yet, there is a concern that the effect of positional encoding may diminish as the attention mechanism is applied repeatedly.

2.3 TSMixer: An All-MLP Architecture for Time Series Forecasting

Recent research [17] has highlighted that simple linear models can be highly effective for time series forecasting, surpassing Transformer-based models, i.e., Autoformer [16], Informer [18] and FEDformer [19]. As illustrated in Fig. 2, TSMixer applies MLPs alternatively in time and feature domains. TSMixer consists of three main components: Time-mixer, Feature-mixer, and Temporal Projection as follows:

Time-Mixer encodes temporal information, e.g., long-term dependencies, into the history X^{-t} by blending across the time direction $X^{-t}_{:c}$ as follows:

$$\begin{aligned} \mathrm{TM}(X^{-t}_{:c}) &= \mathrm{Drop}\Big(\sigma\big((X^{-t}_{:c})^\top W_{\mathrm{TM}} + \boldsymbol{b}_{\mathrm{TM}}\big)\Big), \\ X^{-t}_{:c} &\leftarrow \mathrm{Norm}\Big(X^{-t}_{:c} + \mathrm{TM}(X^{-t}_{:c})^\top\Big), \end{aligned} \tag{7}$$

where $W_{\mathrm{TM}} \in \mathbb{R}^{L\times L}$ and $\boldsymbol{b}_{\mathrm{TM}} \in \mathbb{R}^{1\times L}$ are trainable weight and bias, respectively. $\sigma(\cdot)$, $\mathrm{Drop}(\cdot)$, and $\mathrm{Norm}(\cdot)$ represent an activation function, i.e., ReLU, a dropout operation, and a normalization operation, i.e., 2D batch normalization applied over the $L\times C$ plane along the batch dimension, respectively. Note that the same time-mixing MLPs are shared across all types of features.

Feature-Mixer encodes information regarding the relationship among different observations, e.g., co-occurrence of observations, into the history X^{-t} by blending across the feature direction $X^{-t}_{t:}$ as follows:

$$\begin{aligned} U_{t:} = \mathrm{Drop}\Big(\sigma\big(X^{-t}_{t:} W_{\mathrm{FM}_1} + \boldsymbol{b}_{\mathrm{FM}_1}\big)\Big),\ \ \mathrm{FM}(X^{-t}_{t:}) &= \mathrm{Drop}\Big(U_{t:} W_{\mathrm{FM}_2} + \boldsymbol{b}_{\mathrm{FM}_2}\Big), \\ X^{-t}_{t:} &\leftarrow \mathrm{Norm}\Big(X^{-t}_{t:} + \mathrm{FM}(X^{-t}_{t:})\Big), \end{aligned} \tag{8}$$

where $W_{\mathrm{FM}_1} \in \mathbb{R}^{C\times H}$ and $W_{\mathrm{FM}_2} \in \mathbb{R}^{H\times C}$ are trainable weights, and $\boldsymbol{b}_{\mathrm{FM}_1} \in \mathbb{R}^{1\times H}$ and $\boldsymbol{b}_{\mathrm{FM}_2} \in \mathbb{R}^{1\times C}$ are trainable biases. $U \in \mathbb{R}^{L\times H}$ represents the hidden variables with the the number H of nodes. Note that the same feature-mixing MLPs are shared across all time steps.

Temporal Projection compresses the L-step history $X^{-t}_{:c}$ to the length of future prediction period, i.e., T-step, using a fully-connected layer as follows:

$$\widehat{X^{t+}_{:c}} = \big((X^{-t}_{:c})^\top W_{\mathrm{TP}} + \boldsymbol{b}_{\mathrm{TP}}\big)^\top, \tag{9}$$

where X^{-t} represents the output of the Mixer Layer, $W_{\mathrm{TP}} \in \mathbb{R}^{L\times T}$ and $\boldsymbol{b}_{\mathrm{TP}} \in \mathbb{R}^{1\times T}$ are trainable weight and bias.

The permutation-sensitive properties of the time-mixing MLPs in TSMixer empower the model to effectively capture the dynamic relationships among observations along the time direction, enhancing the prediction performance for multivariate time series data. On the other hand, a limitation exists in the feature-mixing MLPs where the same MLPs are used across time direction, and thus, the identical transformation is applied to vector $X^{-t}_{t:}$ regardless of position in the sequence. This permutation-invariant feature mixing avoids capturing significant relationships between observations, e.g., co-occurrence of observations related to trend and seasonal cycles, and potentially degrades the prediction performance.

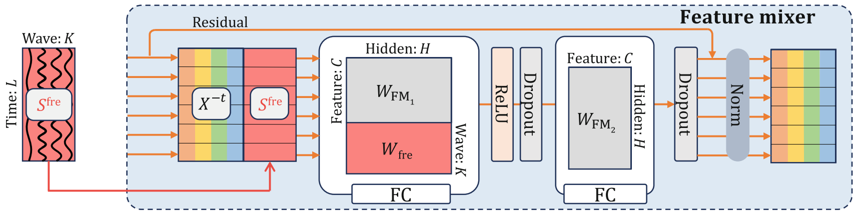


Fig. 3. Architecture of frequency-aware mixer (FAM), a permutation-sensitive extension of Feature Mixing MLPs in TSMixer (in Fig. 2). A matrix $S^{\text{fre}} \in \mathbb{R}^{L\times K}$ contains K different waveforms along the time axis, linearly integrated by weight W_{fre}.

3 Proposed Method

To allow for time-dependent and adaptive feature mixing, we propose to enhance the feature mixing by introducing principal frequency components, called Frequency-Aware Mixer (FAM), and the distance to the principal time step, called Event Proximity-Aware Mixer (EPAM), as additional information vectors.

3.1 Frequency-Aware Mixer (FAM)

We propose Frequency-Aware Mixer (FAM) which incorporates an adaptive frequency component into the first layer of feature-mixing MLPs (in Eq. 8 and Fig. 2) as depicted in Fig. 4 and as follows:

$$\begin{aligned} U_{t:} &= \text{Drop}\Big(\sigma\big(X_{t:}^{-t}W_{\text{FM}_1} + \underline{S_{t:}^{\text{fre}}W_{\text{fre}}} + \boldsymbol{b}_{\text{FM}_1}\big)\Big), \\ S_{tk}^{\text{fre}} &= a_k \cos\left(\frac{2\pi p_k}{N_{\text{tr}}}t\right) + b_k \sin\left(\frac{2\pi p_k}{N_{\text{tr}}}t\right), \quad k = 1, 2, \ldots, K, \end{aligned} \tag{10}$$

where the frequency component $S_{t:}^{\text{fre}}W_{\text{fre}}$ is a linear integration of K different waveforms along time-axis, $S^{\text{fre}} \in \mathbb{R}^{L\times K}$ using weight $W_{\text{fre}} \in \mathbb{R}^{K\times H}$. a_k, b_k, and p_k are trainable parameters tuning the amplitude of cos and sin waves and the frequency for the k-th waveform, respectively.

To prepare initial waveforms $S_{t:}^{\text{fre}}$ representing training time-series data $\mathcal{D}_{\text{tr}}$, we apply a Fourier transform to the sequence $[X_{0c}, X_{1c}, \ldots, X_{(N_{\text{tr}}-1)c}]$ of each observation c and extract $N_{\text{tr}}/2$ waves. Among the waves whose periods do not exceed the history length L, we select m waves with the largest power spectra and set their amplitudes and frequencies as the initial values of a, b, and p for each observation c—there are total $K = mC$ waves.

The feature-mixing in FAM is sensitive to permutations as the frequency components may vary with the time-step t. This allows for flexible feature mixing based on the inherent periodic characteristics of time series, e.g., seasonality, trend cycles, and cyclical cycles. This could potentially enhance the capacity of the model to capture the temporal dynamics of the data and improve the prediction performance.

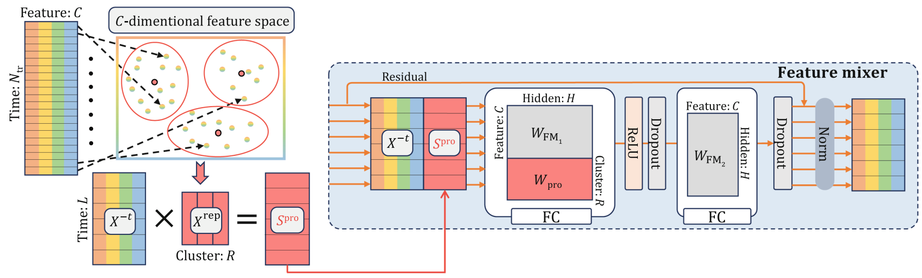


Fig. 4. Architecture of event proximity-aware mixer (EPAM), a temporal characteristics-sensitive extension of Feature Mixing MLPs in TSMixer (in Fig. 2). A matrix $S^{\text{pro}} \in \mathbb{R}^{L \times R}$ contains R different representative observations (events) along the time axis, linearly integrated by weight W_{pro}.

3.2 Event Proximity-Aware Mixer (EPAM)

We propose Event Proximity-Aware Mixer (EPAM) which incorporates an adaptive proximity component into the first layer of feature-mixing MLPs (in Eq. 8 and Fig. 2) as depicted in Fig. 4 and as follows:

$$\begin{aligned} U_{t:} &= \text{Drop}\Big(\sigma\big(X_{t:}^{-t} W_{\text{FM}_1} + \underline{S_{t:}^{\text{pro}} W_{\text{pro}}} + \boldsymbol{b}_{\text{FM}_1}\big)\Big), \\ S_{t:}^{\text{pro}} &= X_{t:}^{-t} X^{\text{rep}}, \end{aligned} \tag{11}$$

where the proximity component $S_{t:}^{\text{pro}} W_{\text{pro}}$ is a linear integration of proximities to R different representative observations (events), $S^{\text{pro}} \in \mathbb{R}^{L \times R}$, using weight $W_{\text{pro}} \in \mathbb{R}^{R \times H}$. $X^{\text{rep}} \in \mathbb{R}^{C \times R}$ is a set of R representative observation vectors and $S_{t:}^{\text{pro}}$ is the inner product (similarity) between an observation vector $X_{t:}^{-t}$ at time-step t and representative vectors X^{rep}.

To prepare representative vectors X^{rep}, we apply a clustering method, e.g., k-means, into C-dimensional vectors across all training time steps, $\{X_{t:}\}_{t=0}^{N_{\text{tr}}-1}$ and set R cluster centroids as X^{rep}.

The feature mixing in EPAM is also sensitive to permutations as the proximity components may vary with the time-step t. This allows for flexible feature mixing based on the natural variability of time series due to the occurrence of various types of events, e.g., holiday and weather events, etc., potentially enhancing the capacity of the model to capture the fluctuation pattern of the data and improve the prediction performance.

3.3 Entire Architecture and Training

The architecture of the proposed method, i.e., $f_{\boldsymbol{\theta}}(X^{-t})$, is a variant of TSMixer depicted in Fig. 2 where its feature mixer component is replaced with our proposed permutation-sensitive feature mixier: FAM (in Fig. 3 or TPAM (in Fig. 4. We refer to the combination of TSMixer with our proposed feature mixers as TSMixer + FAM and TSMixer + TPAM, respectively.

For training the entire architecture, we used mean squared error (MSE) as the loss function $\mathcal{L}(\cdot)$ in Eq. 3 as follows:

$$\mathcal{L}(X^{t+}, f_{\boldsymbol{\theta}}(X^{-t})) = \frac{1}{TC}\left\|X^{t+} - f_{\boldsymbol{\theta}}(X^{-t})\right\|_{\mathrm{F}}^2, \tag{12}$$

where $\|\cdot\|_{\mathrm{F}}$ is Frobenius norm.

We use early stopping with 5-epoch patience based on the validation loss computed using the validation data described in Table 1 and select the best model with the minimum validation loss.

4 Experimental Evaluation

In this section, we show the effectiveness of the proposed method through experiments on seven popular multivariate long-term forecasting benchmarks such as weather, electricity, and traffic.

4.1 Setting and Comparative Methods

We set the length of history observations as $L = 512$ following the work [11], and the length of future prediction observations as $T \in \{96, 192, 336, 720\}$.

We compared the performance of prediction with the state-of-the-art multivariate time series forecasting methods: Transformer-based and MLP-mixer-based models as follows:

- Transformer-based models: we used codes with default settings provided in following githubs:
  - Autoformer [16]: https://github.com/thuml/Autoformer
  - Informer [18]: https://github.com/zhouhaoyi/Informer2020
  - PatchTST [11]: https://github.com/yuqinie98/PatchTST
- MLP-mixer-based models:
  - TSMixer [6]: we used the basic version of TSMixer provided in the github https://github.com/google-research/google-research/tree/master/tsmixer and settings described in the work [6].
  - TMix-Only: we eliminated the feature mixer component (in Fig. 2) from the above TSMixer following the work [6].
  - TSMixer + FAM (proposed method, Sect. 3.1): we set the number of waves for each observation type as $m = 3$ for datasets with fewer observation types, i.e., ETT and Weather, and $m = 1$ for datasets with more types, i.e., Electricity and Traffic. We utilized numpy.fft.rfft function for the implementation of Fourier transform. Other settings are same as TSMixer. In addition, we applied reversible instance normalization (RevIN) into the each input X^{-t} and output $\widehat{X^{t+}}$ of the model [9].
  - TSMixer + EPAM (proposed method in Sect. 3.2): we set the number of representative observations as $R = 5$. We utilized sklearn.cluster module for k-means clustering. Other settings are same as TSMixer + FAM.

Table 1. Details of the datasets used in the experiments

	ETTh1/h2	ETTm1/m2	Weather	Electricity	Traffic
No. of obs. C	7	7	21	321	862
Time steps	17,420	69,680	52,696	26,304	17,544
Time cycle	1 h	15 min	10 min	1 h	1 h
Data split train:valid:test	12:4:4 [month]		70:10:20 [%]		

4.2 Datasets

We used seven real-world multi-variate time series datasets: ETT [13], Weather [3], Electricity [2], and Traffic [1], provided by the work of Autoformer [16] in https://github.com/thuml/Autoformer.

More specifically, Electricity Transformer Temperature (ETT) datasets contain two-year sequences of loads and oil temperature collected from electricity transformers every 1 h and 15 min. Weather dataset contains one-year sequences of 21 meteorological indicators, such as air temperature and humidity, recorded every 10 min. Electricity dataset contains three-year sequences of electricity consumption of 321 customers, collected every hour. Finally, Traffic dataset contains two-year sequences of road occupancy rates at 862 different places, recorded every hour. Table 1 summarizes the details of the datasets.

As a preprocessing step for the data, we divided the sequences from each dataset into training, validation, and test subsequences, as described in Table 1. We then calculated the mean and standard deviation for each subsequence $[X_{0c}, X_{1c}, \ldots]$ for each observation c and sed these values to normalize the corresponding subsequences. Then, we used these normalized subsequences as training $\mathcal{D}_{\rm tr}$ (in Eq. 4), validation $\mathcal{D}_{\rm val}$, test $\mathcal{D}_{\rm te}$ (in Eq. 5) data.

4.3 Result

The experimental results are shown in Table 2. The performance is measured using MSE of test data $\mathcal{D}_{\rm te}$ as follows:

$$\mathrm{MSE}(\mathcal{D}_{\rm te}) = \frac{1}{TC|\mathcal{D}_{\rm te}|} \sum_{(X^{-t}, X^{t+}) \in \mathcal{D}_{\rm te}} \left\| X^{t+} - f_{\boldsymbol{\theta}}(X^{-t}) \right\|_{\rm F}^2. \tag{13}$$

In principle, multivariate models that simultaneously consider the relationships between time and features are expected to offer higher flexibility and performance in time series forecasting compared to univariate models, which only independently consider the time series of individual features. However, Table 2 demonstrates that the performance of Autoformer and Informer in multivariate models is inferior to that of the univariate model. Furthermore, the performance of TSMixer is equivalent to that of TMix-Only, which lacks a feature mixer, indicating that the co-occurrences in feature direction provided by a feature mixer, are not necessarily important for prediction, as reported in the work [6].

Table 2. Performance comparison in multivariate time series forecasting. The performance is measured using the MSE computed from each test data $\mathcal{D}_{\text{te}}$ (in Eq. 13). Those performance surpassing TSMixer is indicated in red among multivariate models. In addition, the best performance among all models is indicated in bold.

Dataset		Univariate Model		Multivariate Model				
Model		TMix-Only	PatchTST	Autoformer	Informer	TSMixer	+FAM	+EPAM
ETTh1	96	0.3643	0.3721	0.6110	0.8437	0.3645	0.3644	**0.3626**
	192	0.3995	0.4106	0.5105	0.9920	0.4011	0.4010	**0.3948**
	336	0.4231	0.4216	0.5767	1.3062	0.4249	0.4245	**0.4104**
	720	0.4543	0.4473	0.6887	1.3983	0.4562	0.4535	**0.4342**
ETTh2	96	**0.2698**	0.2749	0.4826	0.3512	0.2730	0.2707	0.2734
	192	0.3366	0.3385	0.5602	0.3805	0.3386	0.3347	**0.3344**
	336	0.3602	**0.3302**	0.7877	0.3832	0.3631	0.3639	0.3654
	720	0.4229	**0.3839**	0.9875	0.6024	0.4219	0.4269	0.4520
ETTm1	96	0.2879	0.2909	0.4769	0.6620	0.2861	0.2858	**0.2852**
	192	0.3256	0.3349	0.5854	0.7321	0.3266	**0.3254**	0.3326
	336	**0.3561**	0.3636	0.6663	0.6024	0.3568	0.3587	0.3691
	720	**0.4160**	0.4166	0.6939	0.6474	0.4167	0.4202	0.4192
ETTm2	96	0.1678	**0.1652**	0.2866	0.3290	0.1677	0.1654	0.1962
	192	**0.2184**	0.2226	0.3288	0.6741	0.2185	0.2222	0.2299
	336	0.2817	**0.2735**	0.3809	0.8493	0.2815	0.2782	0.2953
	720	0.4015	**0.3593**	0.4683	0.9752	0.4175	0.4074	0.5512
Weather	96	0.1491	0.1482	0.3799	0.4160	0.1489	**0.1474**	0.1480
	192	**0.1888**	0.1938	0.3174	0.7115	0.1894	0.1897	0.1913
	336	0.2396	0.2468	0.3487	0.9766	0.2370	0.2429	**0.2363**
	720	0.3148	0.3136	0.3888	1.1191	0.3124	0.3219	**0.3087**
Electricity	96	0.1307	0.1289	0.2265	0.3316	0.1300	**0.1288**	0.1290
	192	0.1497	0.1468	0.2203	0.3574	0.1487	0.1491	**0.1460**
	336	0.1636	0.1659	0.2203	0.3848	0.1633	0.1633	**0.1604**
	720	0.1940	0.1997	0.2441	0.4170	0.1948	0.1939	**0.1925**
Traffic	96	0.3795	0.4104	0.6804	1.2582	0.3798	**0.3758**	0.3791
	192	0.3979	0.4125	0.6732	1.3366	0.3982	**0.3936**	0.3959
	336	0.4152	0.4232	0.7119	1.4528	0.4147	**0.4085**	0.4171
	720	0.4509	0.4614	0.7401	1.4859	0.4497	**0.4487**	0.4498

On the other hand, Table 2 shows that the proposed methods, TSMixer+FAM and TSMixer+EPAM, which enhance the feature-mixer with permutation sensitivity, outperform TSMixer in various datasets and future prediction steps, i.e., T. This indicates the potential of the proposed methods for permutation-dependent feature mixing in adaptively modeling relationships between features and reveals the potential importance of feature mixing in multivariate time series forecasting.

Table 3. Comparison of parameter counts and averaged inference time per instance, measured using the test data $\mathcal{D}_{\text{te}}$ in Traffic dataset.

Traffic		
Model	# of params	Average inference time (ms)
TMix-Only	903,292	1.557
TSMixer	1,795,112	1.838
+FAM	2,245,948	3.769
+EPAM	1,797,722	1.831

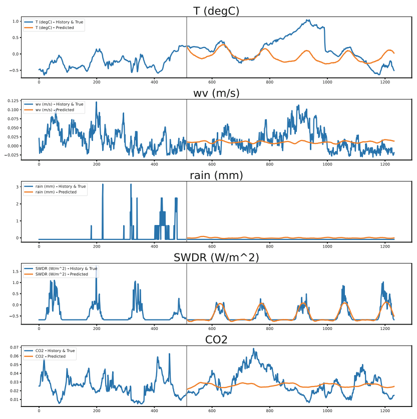
Table 3 presents a comparison of parameter counts and the average time per inference, measured using the test data $\mathcal{D}_{\text{te}}$ in Traffic dataset with the most types of observations as shown in Table 1. As the table shows, multivariate models tend to have larger parameters as the number of observations C increases compared to univariate models. Furthermore, in the case of the TSMixer+FAM, model, datasets with a large number of observations result in a significantly higher number of extracted frequencies K (in Eq. 10), leading to a much longer inference time compared to other models. In the future, it will be necessary to adopt strategies such as feature grouping to reduce the number of additional vectors while maintaining predictive accuracy.

4.4 Analysis

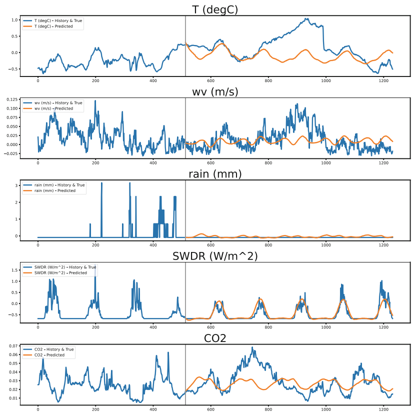
In addition, Fig. 5 depicts examples of forecasts by TSMixer, FAM, and EPAM, for T (Temperature), WV (Wind Velocity), rain(precipitation), SWDR (Short Wave Downward Radiation), and CO2 (CO2 concentration) of Weather dataset with forecasting length $T = 720$—the history $X^{-t}_{:c}$ and ground truth $X^{t+}_{:c}$ (Eq. 1) in blue and the prediction $\widehat{X^{t+}}_{:c}$ (in Eq. 2) in orange.

Figure 5 shows that compared to TSMixer, TSMixer+FAM is able to more closely follow the true values with its periodic predictions such as in WV and CO2. This outcome underscores the impact of FAM's dynamic frequency-dependent feature mixing, which is further enhanced by the incorporation of additional frequency information into the feature mixing process. Therefore, in fields such as multivariate time series forecasting involving frequency components, like temporal traffic patterns, FAM's ability to accurately track waveforms is considered effective.

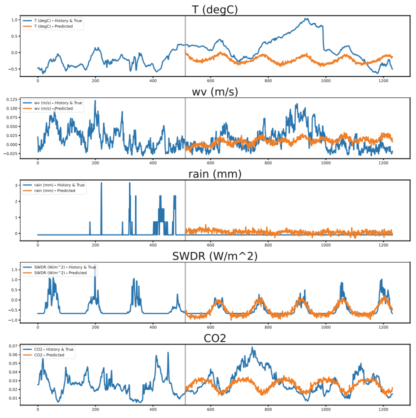
Similarly, EPAM is able to more closely follow the true values with its finer-grained predictions. This fine-grained prediction reflects the impact of EPAM's approach of integrating distance information between major events and each historical point into the feature mixing. Therefore, in multivariate time series



(a) TSMixer



(b) TSMixer + FAM



(c) TSMixer + EPAM

Fig. 5. Examples of ground truth (in blue) and prediction (in orange) by TSMixer, +FAM, and +EPAM models, for T (Temperature), RH (Relative Humidity), WV (Wind Velocity), SWDR (Short Wave Downward Radiation), and CO2 (CO2 concentration) variables in the test data $\mathcal{D}_{\text{tr}}$ of Weather dataset with forecasting length $T = 720$. The history length L is fixed at 512 for all experiments, and the grey vertical line indicates the boundary date between the history and the future. (Color figure online)

forecasting involving rapid changes within short periods, such as Electricity data, EPAM's ability to make detailed adjustments is considered effective.

From these observations, it is clear that in the domain of multivariate time series data with complex inter-feature relationships, the proposed method can enhance the ability to utilize the relationships between features for more accurate forecasting.

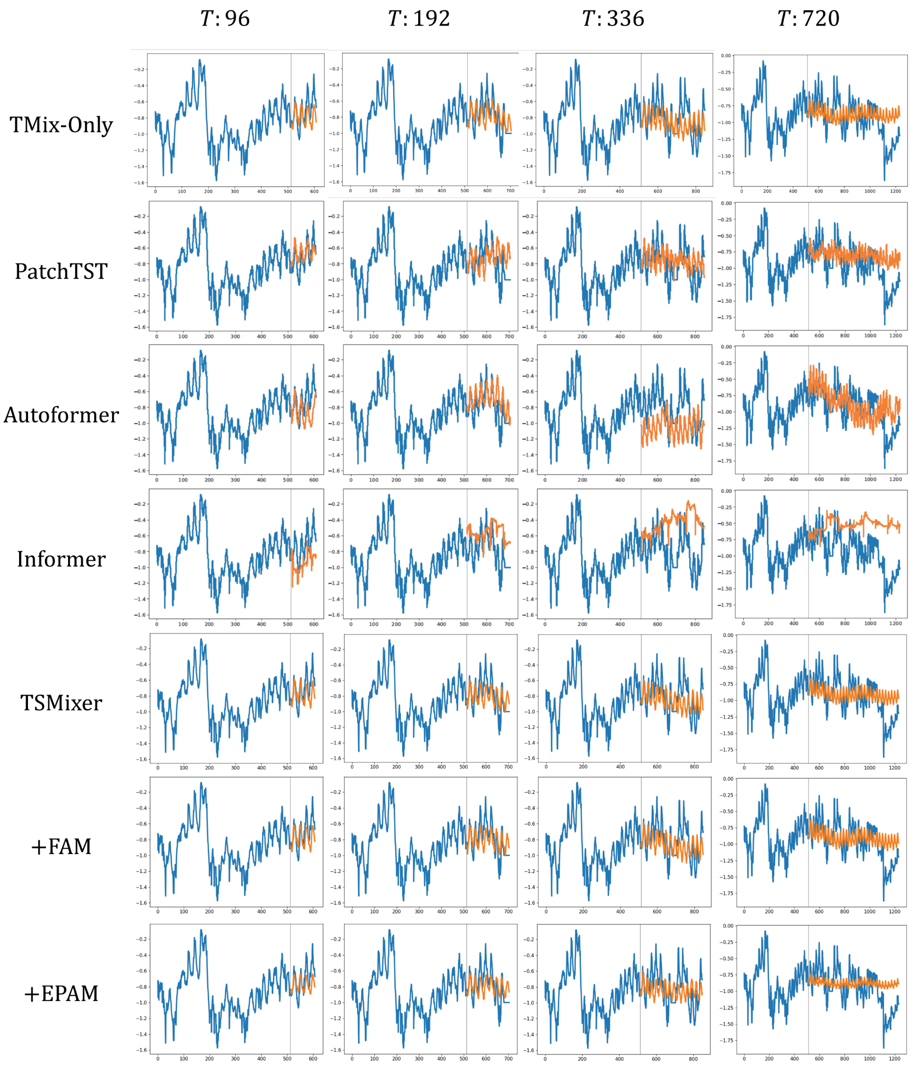


Fig. 6. Examples of ground truth (in blue) and prediction (in orange) for Oil Temperature (OT) variable in the test data $\mathcal{D}_{\text{tr}}$ of ETTh1 dataset. Each row and column corresponds to a different model and forecast length $T \in \{96, 192, 336, 720\}$. The history length L is fixed at 512 for all experiments, and the grey vertical line indicates the boundary date between the history and the future. (Color figure online)

To further analyze the effectiveness of the proposed method, we visualized the experimental results for different forecasting lengths across all comparison methods. Fig. 6 depicts examples of forecasts for Oil Temperature (OT) variable of ETTh1 dataset with forecasting length $T \in \{96, 192, 336, 720\}$—the history

$X_{:c}^{-t}$ and ground truth $X_{:c}^{t+}$ (Eq. 1) in blue and the prediction $\widehat{X^{t+}}_{:c}$ (in Eq. 2) in orange, where c corresponds to OT.

In contrast to nonlinear multivariate models, i.e., Autoformer and Informer, which increasingly diverge from the ground truth as the forecast horizon extends, TSMixer-based models are capable of delivering forecasts that are on par with univariate models across various forecast lengths, T. Furthermore, compared to TSMixer, FAM tends to predict with more periodic trends, while EPAM tends to predict with finer amplitudes. This likely occurs because the models have effectively adjusted the mixing features based on the overall frequency of the training data and the proximity to representative events. Consequently, these models not only capture the temporal mixing of information but also extract useful information through feature mixing.

5 Conclusion

In this study, we enhance feature mixing in the TSMixer model for multivariate time series forecasting by introducing principal frequency components through the Frequency-Aware Mixer (FAM) and incorporating distances to principal time steps with the Event Proximity-Aware Mixer (EPAM). These proposed methods enable time-dependent and adaptive feature mixing, demonstrating the utility of permutation-dependent feature mixing for dynamically capturing the relationships between features. Experimental results across various real-world datasets indicate the potential of the proposed methods for permutation-dependent feature mixing in adaptively modeling relationships between features and reveal the importance of feature mixing in multivariate time series forecasting.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. California Department of Transportation. https://pems.dot.ca.gov/. Accessed 23 Mar 2023
2. ElectricityLoadDiagrams. https://archive.ics.uci.edu/ml/datasets/ElectricityLoadDiagrams20112014. Accessed 23 Mar 2023
3. Max-Planck-Institut für Biogeochemie - Wetterdaten. https://www.bgc-jena.mpg.de/wetter/. Accessed 23 Mar 2023
4. Box, G.E.P., Jenkins, G.M.: Time Series Analysis, Forecasting and Control (1970)
5. Box, G.E., Jenkins, G.M.: Some recent advances in forecasting and control (1968)
6. Chen, S.A., Li, C.L., Yoder, N.C., Arık, S.O., Pfister, T.: Tsmixer: an all-mlp architecture for time series forecasting (2023). https://arxiv.org/abs/2303.06053
7. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. (1997)
8. Khabbazan, S., et al.: Crop monitoring using sentinel-1 data: a case study from The Netherlands (2019)
9. Kim, T., Kim, J., Tae, Y., Park, C., Choi, J.H., Choo, J.: Reversible instance normalization for accurate time-series forecasting against distribution shift (2022). https://openreview.net/forum?id=cGDAkQo1C0p

10. Lai, G., Chang, W.C., Yang, Y., Liu, H.: Modeling long- and short-term temporal patterns with deep neural networks. In: SIGIR (2018)
11. Nie, Y., Nguyen, N.H., Sinthong, P., Kalagnanam, J.: A time series is worth 64 words: long-term forecasting with transformers. In: International Conference on Learning Representations (2023)
12. Rangapuram, S.S., Seeger, M.W., Gasthaus, J., Stella, L., Wang, Y., Januschowski, T.: Deep state space models for time series forecasting. Adv. Neural Inf. Process. Syst. **31** (2018)
13. THUML: ETDataset: GitHub Repository. https://github.com/zhouhaoyi/ETDataset (2023). Accessed 22 Mar 2023
14. Vaswani, A., et al.: Attention is all you need. Adv. Neural Inf. Process. Syst. **30** (2017). https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf
15. Wen, R., Torkkola, K., Narayanaswamy, B., Madeka, D.: A multi-horizon quantile recurrent forecaster. In: NeurIPS (2017)
16. Wu, H., Xu, J., Wang, J., Long, M.: Autoformer: decomposition transformers with auto-correlation for long-term series forecasting. Adv. Neural Inf. Process. Syst. (2021)
17. Zeng, A., Chen, M., Zhang, L., Xu, Q.: Are transformers effective for time series forecasting? (2022). https://arxiv.org/abs/2205.08459
18. Zhou, H., et al.: Informer: beyond efficient transformer for long sequence time-series forecasting. In: Proceedings of the AAAI Conference on Artificial Intelligence (2021). https://doi.org/10.48550/arXiv.2012.07436
19. Zhou, T., Ma, Z., Wen, Q., Wang, X., Sun, L., Jin, R.: Fedformer: frequency enhanced decomposed transformer for long-term series forecasting (2022)