ISDformer: Iso-Spatiotemporal Decoupling for Long-Term Time Series Forecasting

Anonymous submission

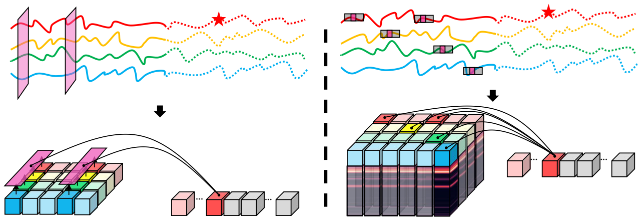
Abstract

Transformer-based models have demonstrated remarkable performance in time series forecasting tasks, leveraging their ability to capture long sequence features. However, existing transformer-based models follow the traditional NLP learning paradigm which treats all variables at the same timestamp as a single entity. This paradigm limits the learning of deep-level asynchronous spatiotemporal relationships among covariates, and can not adequately address the random fluctuation and inflection point challenges. In this paper, we propose an iso-spatiotemporal decoupling model ISDformer to tackle the critical issues for time series forecasting. Our model has three distinctive characteristics. First, in the embedding stage, we propose F&V embedding to decouple spatiotemporal relationships and independently embed multi-dimensional variables at the same timestamp. We also introduce the frequency domain features to mitigate the impact of random fluctuations and inflection points. Second, in the encoder stage, we propose vectors enhancement attention (VEA) to capture related information at synchronous and asynchronous timestamps. Third, in the decoder stage, we propose a spatiotemporal cross-aggregation attention (SCA) that combines historical sequence features obtained from the encoder stage. Extensive experiments on five large-scale datasets show that ISDformer outperforms the existing methods. The source code will be open sourced.

Introduction

Time series forecasting refers to using existing data arranged in a sequence according to time, modeling and analyzing its changing trends and magnitude, to make forecastings about the levels that may be reached in future periods. Time series forecasting plays an important role in decision-making in various fields such as energy (Demirel et al. 2012), transportation (Barros, Araujo, and Rossetti 2015), weather (Biswas et al. 2014; Angryk et al. 2020), and economics (Patton 2013; Ding et al. 2015).

With the development of deep learning, deep neural networks (DNN) have been widely used for sequence feature modeling. Especially, transformer-based models have achieved state-of-the-art forecastive performance in the field of time series forecasting due to their excellent ability to capture features of long sequences (Wen et al. 2022; Zhou et al. 2022; Wu et al. 2021, 2023; Zhou et al. 2021; Li et al. 2019; Wu et al. 2020). Analyzing the current popular transformer-based methods, we found that they follow the common practices inherited from the NLP learning paradigm and have significant effects in capturing and learning explicit sequence relationships such as periods and frequency domains, as shown in figure 1 (a). Nevertheless, when it comes to capturing intricate asynchronous spatiotemporal relationships and mitigating fortuitous fluctuations, these approaches exhibit limitations in pattern acquisition, mainly reflected in the following two aspects:


(a) Typical transformer

(b) ISDformer

Figure 1: (a) Illustration of synchronous multidimensional variable coupling learning. (b) llustration of cross dimensional asynchronous spatiotemporal decoupling learning combined with frequency domain.

The coupling embedding of synchronous multidimensional variables restricts the expression of asynchronous spatiotemporal relationships. In the embedding stage, transformer-based methods embed different variables into a word vector at the same time, but predict that the target variable is independently affected by different features from previous moments. If we simply measure the impact of features from different dimensions on the predicted values at different times by coupling them to the same dimension, this will result in the model losing its ability to represent composite effects. In addition, this pointwise direct encoding method ignores the randomness of time series data, making the attention mechanism unable to detect the negative impact of outliers on feature expression during the operation process.

Synchronous multi-dimensional variable coupling decoding constraints asynchronous spatiotemporal feature learning. In the encoding stage, the aggregation operation of self-attention in the transformer encoder ensures that each vector has a contextual relationship. However, the model should be more concerned about which information can affect the predicted value, and the learning of this contextual relationship becomes cumbersome and worthless; on the other hand, in the aggregation and prediction steps of the decoder, 1-dimensional aggregation methods are usually used, and it is impossible to use captured 2-dimensional asynchronous spatiotemporal relationships to obtain better predicted values.

In this paper, we propose an iso-spatiotemporal decoupling model ISDformer for time series forecasting. In the embedding stage, we propose F&V embedding, which encodes each variable at the same moment separately to achieve the purpose of iso-spatiotemporal decoupling, and introduces frequency domain information to enhance the model's suppression of accidental volatility. In the encoder stage, we propose the vectors enhancement attention (VEA) to decouple and process useful synchronous and asynchronous spatiotemporal information for forecasting. In the decoder stage, we propose the spatiotemporal cross aggregation (SCA) to aggregate the features obtained in the encoder stage from both temporal and spatial dimensions, and finally obtain forecasting results.

This paper makes **three major contributions**:

1 We find the importance of covariates for forecast targets in multivariate time variable forecasting problems and propose a spatiotemporally decoupled embedding method.

2 We propose ISDformer, an iso-spatiotemporal decoupling time series forecasting model that can capture potential relationships between asynchronous spatiotemporal variables and make forecastings based on them.

3 We conducted experiments on five datasets from various fields, and the results show that ISDformer has significantly outperforms existing methods.

Related Work

Pre-Transformer Methods. Early methods were mainly based on statistical learning techniques. Statistical models represented by ARIMA (Ho, Xie, and Goh 2002) and SARIMAX (Elamin and Fukushige 2018; Tarsitano and Amerise 2017) can learn the linear trend of time series. Machine learning methods based on SVM (Thissen et al. 2003; Sapankevych and Sankar 2009), GBRT (Yang et al. 2018), and Hidden Markov models further enhance the learning ability of high-dimensional time series (Li et al. 2021; Cheng and Li 2011; Hanif et al. 2017). Later, (Galicia et al. 2019) integrated the idea of ensemble learning, combining decision trees, gradient boosting trees, and random forests into three models.

With the development of deep learning, modeling based on deep learning has been applied to time series forecasting tasks. In particular, the time characteristic learning ability based on RNN (Zhang and Man 1998) and LSTM (Hochreiter and Schmidhuber 1997; Ma et al. 2015; Bouktif et al. 2020) has achieved better performance in many short-term forecasting tasks. However, due to the recursive nature of recurrent neural networks, it is easy to accumulate errors, which limits its ability to extract deep spatiotemporal features in long sequences. Therefore, there is a bottleneck in long-term sequence prediction problems.

Transformer-Based Methods. Transformer was initially proposed for NLP tasks and later extended to multi-class visual tasks (Vaswani et al. 2017; Khan et al. 2022). Due to the similarity between language sequences and time sequences, this work was quickly extended to the field of time series prediction (Wu et al. 2020) and achieved better prediction performance compared to traditional deep learning models based on CNN (Borovykh, Bohte, and Oosterlee 2017; Liu et al. 2022) and RNN (Salinas et al. 2020). Subsequently, a large number of transformer-based time series prediction works have conducted in-depth research on how to reduce the computational overhead of the attention mechanism and how to model the relationship of the time dimension more reasonably.

In terms of computational overhead optimization, methods such as Logtrans (Li et al. 2019), Informer (Zhou et al. 2021), etc., which mainly explore sparse forms of attention mechanisms, have significantly reduced computational complexity; in terms of time dimension modeling, TimesNet (Wu et al. 2023), Autoformer (Wu et al. 2021), and FEDformer (Zhou et al. 2022) further enhance the characterization and learning ability of the time dimension by learning multi-periods relationships and mining sequence frequency domain features. However, this class of transformer-based methods basically follows the original learning paradigm for processing NLP, coupling multi-dimensional variables at the same moment, resulting in ineffective learning of asynchronous spatio-temporal variable relationships. Crossformer (Zhang and Yan 2022) focuses on the characterization of the relationship between covariates, by embedding variables separately after sliding slices along the time axis, and designing two-stage attention in the time dimension and space dimension to provide inspiration for asynchronous spatio-temporal relationship learning in time series.

Methodology

The time series forecasting task is defined as: for a given sequence $\mathbf{X}_t,\ t \in [1, \mathrm{T}], \mathbf{X} \in \mathbb{R}^{\mathrm{T}\times\mathrm{D}}$, where T is the historical time step and D is the dimension of the features at each time step, model its intrinsic features and forecast the values at future τ timestamps: $\mathbf{X}_{\mathrm{T}+t},\ t \in [1, \tau], \mathbf{X} \in \mathbb{R}^{\tau\times\mathrm{D}}$.

To address the problem raised in the introduction, we propose ISDformer, which designs an Embedding, Encoder, and Decoder suitable for capturing asynchronous spatiotemporal relationships, as shown in Figure 2. Detailed discussions of each part are given in the following text.

F&V Embedding

In previous time series forecasting work, for a given time series: $\mathbf{x}_t^d \in \mathbb{R}^{1\times1}, 1 \leq d \leq \mathrm{D}, 1 \leq \mathrm{t} \leq T$, where D represents the dimension of the variable and T represents the length of the given time series, we have $\mathbf{h}_t \in \mathbb{R}^{1\times\text{model depth}}$. specifically,

$$\mathbf{h}_t = \mathrm{temp}(\mathbf{date}) + \mathrm{conv1D}(\mathbf{x}_t) + \mathrm{pos}(\mathbf{x}_t), \qquad (1)$$

where $\mathbf{x}_t = \mathrm{concat}\left[\mathbf{x}_t^1, \mathbf{x}_t^2, ..., \mathbf{x}_t^{\mathrm{D}}\right]$. In the equation, conv1D represents a 1-dimensional convolution operation applied along the time axis, with D input features and model

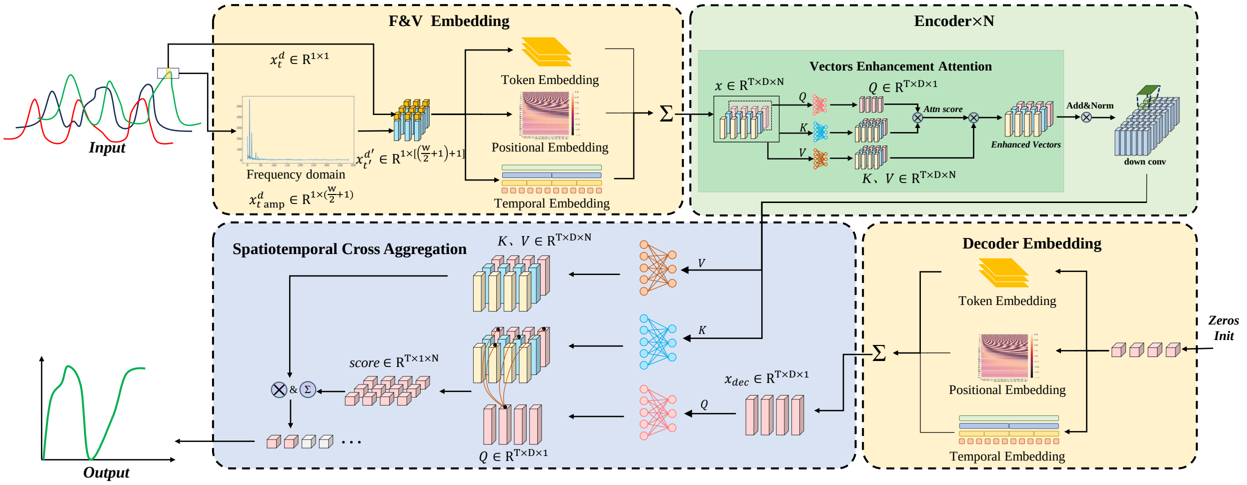

F&V Embedding
$x_t^d \in \mathrm{R}^{1\times1}$
Input
Frequency domain
$x_{t\,\mathrm{amp}}^d \in \mathrm{R}^{1\times(\frac{w}{2}+1)}$
$x_{t'}^{d'} \in \mathrm{R}^{1\times[(\frac{w}{2}+1)+1]}$
Token Embedding
Positional Embedding
Temporal Embedding
Σ
Encoder×N
Vectors Enhancement Attention
$x \in \mathrm{R}^{T\times D\times N}$
$Q \in \mathrm{R}^{T\times D\times 1}$
Q
K
V
Attn score
Enhanced Vectors
Add&Norm
down conv
$K、V \in \mathrm{R}^{T\times D\times N}$
Spatiotemporal Cross Aggregation
$K、V \in \mathrm{R}^{T\times D\times N}$
V
K
Q
$score \in \mathrm{R}^{T\times 1\times N}$
& Σ
$x_{dec} \in \mathrm{R}^{T\times D\times 1}$
$Q \in \mathrm{R}^{T\times D\times 1}$
Output
Decoder Embedding
Token Embedding
Positional Embedding
Temporal Embedding
Σ
Zeros Init

Figure 2: ISDformer model overview. The time series data stream is learned using a cross-dimensional decoupling paradigm.

depth output features. The term "pos" refers to the positional embedding, while "temp" refers to the encoding of date features. This embedding approach was initially employed in the Informer (Zhou et al. 2021) model, originating from the field of NLP and later adapted for time series forecasting. As mentioned in the introduction, this encoding method couples variables of different dimensions, making it difficult for the model to mine asynchronous spatio-temporal relationships between variables and to distinguish those outliers.

To solve these problems, we proposed F&V embedding, an improved 2-dimensional word embedding method suitable for time series prediction. Firstly, in order to achieve the purpose of decoupling word vectors in space, we regard each variable at the same moment as an isolated individual and different variables at different times still maintain their sequential relationship:$\mathbf{x}_t^d, d \in [1, \mathrm{D}]$ stands for dimension, $t \in [1, \mathrm{T}]$ stands for time.

Then, in order for the model to feel the situation of an isolated point in a sequence, we need to obtain its frequency distribution information. Firstly take several points before and after that point together, window them for Fourier decomposition to obtain their frequency domain information; then transform the decomposition result into amplitude and phase form and discard its phase while only retaining amplitude information:

$$\mathbf{x}_{t\,\mathrm{amp}}^d = \mathrm{ABS}(\mathrm{FFT}(\mathrm{win}(\mathbf{x}_{t-\frac{\mathrm{w}}{2}}^d, \ldots, \mathbf{x}_t^d, \ldots, \mathbf{x}_{t+\frac{\mathrm{w}}{2}}^d))), \quad (2)$$

where $\mathbf{x}_{t\,\mathrm{amp}}^d \in \mathbb{R}^{1\times(1+\frac{\mathrm{w}}{2})}$. ABS here stands for taking the absolute value of the result obtained from the Fast Fourier Transform (FFT), and $\mathbf{x}_{t\,\mathrm{amp}}^d$ is the amplitude of the frequency component. In order to maintain consistent magnitudes, it is common practice to normalize the amplitude as well.Then concatenate variables at each moment and each dimension with frequency domain amplitude obtained from decomposition to obtain word vectors to be embedded:

$$\mathbf{x'}_t^d = \mathrm{concat}(\mathbf{x}_t^d, \mathbf{x}_{t\,\mathrm{amp}}^d), \quad (3)$$

where $\mathbf{x'}_t^d \in \mathbb{R}^{1\times(2+\frac{\mathrm{w}}{2})}$. We refer to the above steps as "FreqEmbedding". For the word embedding, we have retained a structure similar to Informer(Zhou et al. 2021), but to preserve the synchronized spatiotemporal relationships of word vectors without coupling, we use a 2-dimensional convolution with a kernel size of 1x3. To ensure result consistency, we need to extend along the time axis. This means that convolution is applied along the time axis, while no processing is done in the spatial domain.

As for positional encoding, we adopted the sine and cosine encoding method from (Vaswani et al. 2017). We generated a 1-dimensional sequence solely for distinguishing the relative order of word vectors along the time axis. This sequence is then added to the word vectors using Python's broadcasting mechanism, as the relative order of spatial features is not crucial for the model.

Our proposed asynchronous spatiotemporal decoupling embedding with frequency domain fusion can be expressed as follows:

$$\mathbf{h}_t^d = \mathrm{Conv2D}(\mathbf{x'}_t^d) + \mathrm{pos}(\mathbf{x'}_t) + \mathrm{temp}(\mathbf{date}), \quad (4)$$

where $1 \le t \le \mathrm{T},\ 1 \le d \le \mathrm{D}, \mathbf{h} \in \mathbb{R}^{\mathrm{T}\times\mathrm{D}\times\mathrm{model\,depth}}$.

Vectors Enhancement Attention

After obtaining the vectors after the feature engineering of the embedding, in order to avoid the redundant contextual information brought by the aggregation operation of the traditional attention mechanism and emphasize the word vectors that are useful for the predicted value, we designed a 2-dimensional vectors enhancement attention (SCA). Similar to the traditional attention mechanism, since we need to know which covariates have an impact on the target variable, first, for the variable that needs to be predicted, we denote it as traget, and generate $\mathbf{Q}$ through a linear layer. For all variables, $\mathbf{K}$ and $\mathbf{V}$ are generated through a linear layer respectively: $\mathbf{Q} = P_Q(\mathbf{target})$, Then use a $\mathbf{Q}$ to query all $\mathbf{K}$s and get attention weights: $\mathbf{K} = P_K(\mathbf{h}_t^d)$, $\mathbf{V} = P_V(\mathbf{h}_t^d)$. For each $\mathbf{Q}$, the 2-dimensional attention weight can be generated. In

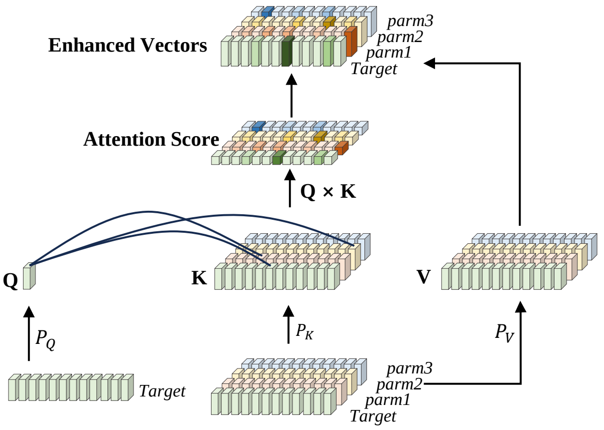


Figure 3: Vectors enhancement attention (VEA) structure.

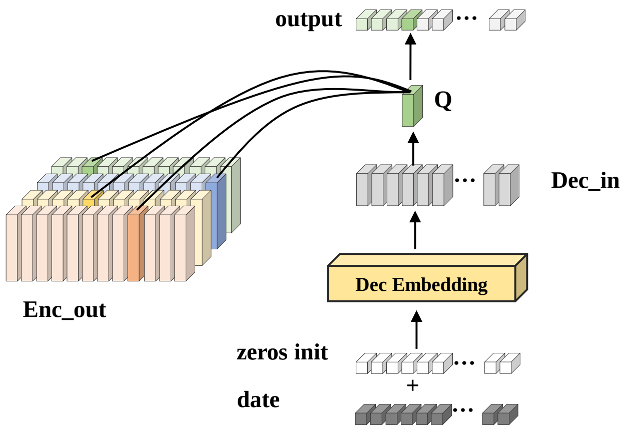


Figure 4: Spatial cross aggregation (SCA) structure.

this way, i groups of attention weights are obtained:

$$\mathbf{Attn}_i = \mathbf{Q}_i \times \mathbf{K}. \tag{5}$$

Add these weights according to the position of **Q** and normalize them to obtain a group of 2-dimensional attention weights. This weight represents the importance of each covariate to the variable to be forecasted:

$$\mathbf{score} = \mathbf{norm}(\frac{1}{\mathrm{sqrt(D)}}\sum_i \mathbf{Attn}_i), \tag{6}$$

In the equation, **norm** refers to performing 2-dimensional normalization on the result. Finally, in order to emphasize these importance degrees, unlike traditional attention aggregation operations, we directly multiply attention weights by **V**, so as to highlight some important features:

$$\textbf{Enhanced Vectors} \;=\; \mathbf{V} \times \mathbf{score}. \tag{7}$$

Its structure is shown in the figure 3.

It is worth noting that since this word vector enhancement mechanism has the same input and output shape size, it can seamlessly replace self-attention mechanisms. And although this mechanism is designed for two-dimensional features, by setting D = 1, it can be applied to situations where there is only one feature.

Spatiotemporal Cross Aggregation

As described in the F&V embedding section, when embedding word vectors, the spatial features have potential connections and therefore cannot be directly coupled and encoded. After obtaining these independent spatial encoding vectors, the next step is to process them to aggregate the features in the most appropriate way. For this purpose, we designed a spatiotemporal cross aggregation module to aggregate 2-dimensional spatiotemporal information.

In the decoder stage, a zero tensor of forecasting length is first initialized, and then it is subjected to word embedding, position embedding, and date embedding to obtain a sequence embedded with position and time information. Since the resulting tensor is 1-dimensional, this process can follow the original embedding process but without conv1D embed, because zero initialization values do not need to be embedded:

$$\mathbf{Dec\_in} = \mathrm{pos}(\mathbf{zeros\ init}) + \mathrm{temp}(\mathbf{date}). \tag{8}$$

For the results of the encoder output, they are passed through two fully connected layers to generate a set of **K** and a set of **V** respectively: $\mathbf{K} = P_K(\mathbf{Enc\_out})$, $\mathbf{V} = P_V(\mathbf{Enc\_out})$. Next, a series of **Q**s are generated through a fully connected layer from the word embedding results: $\mathbf{Q} = P_Q(\mathbf{Dec\_in})$, and these **Q**s are used to synchronously query the spatiotemporal features generated during the encoder stage to obtain a series of attention scores:

$$\mathbf{Attn}_j = \mathbf{Q}_j \times \mathbf{K}, \tag{9}$$

where $\mathbf{Attn}_j \in \mathbb{R}^{T'\times \mathrm{D} \times 1}$. The T' is not same with T which refered before, as the result of down conv showed in figure 2. Finally, all features are aggregated using attention scores to obtain the output result:

$$\mathbf{output}_j = \mathbf{Attn}_j \times \mathbf{V}, \tag{10}$$

where $\mathbf{output}_j \in \mathbb{R}^{1\times \mathrm{d\_model}}$, $j \in [1, \tau]$. In this way, compared with the classic cross-attention mechanism, this method extends attention from 1-dimension to 2-dimensions, further enriching the feature mining and expression methods of attention mechanisms, bringing more flexible expression effects.

Experiment

Experimental Agreement

Dataset. We trained and tested our model on 5 real datasets. The experimental datasets are described as follows:

- ETTm2 (Zhou et al. 2021): Contains transformer oil temperature data from July 1, 2016 to June 26, 2018, recorded every 15 minutes, with a total of 69,680 records. The dataset contains 7 attributes and forecasts the target attribute, oil temperature (OT).
- Traffic (PeMS 2014): Contains road occupancy rates measured by California highway sensors from July 1,

Methods		Informer		Transformer		Autoformer		FEDformer		TimesNet		**ISDformer**	
Sources		(Zhou et al. 2021)		(Wu et al. 2020)		(Wu et al. 2021)		(Zhou et al. 2022)		(Wu et al. 2023)		**Ours**	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTm2	96	0.265	0.396	**0.259**	0.396	0.350	0.434	0.320	0.435	0.260	**0.372**	0.593	0.618
	384	0.284	0.413	0.291	0.416	0.370	0.473	0.330	0.452	**0.272**	**0.391**	0.625	0.668
	672	0.302	**0.404**	**0.290**	0.411	0.396	0.505	0.403	0.449	0.296	0.429	0.654	0.689
	1344	0.458	0.490	0.426	0.539	0.605	0.584	0.663	0.796	**0.352**	**0.464**	0.872	0.858
Traffic	96	0.311	0.378	0.229	0.318	0.226	0.298	0.263	0.364	0.129	0.216	**0.052**	**0.163**
	384	0.312	0.386	0.246	0.321	0.240	0.345	0.245	0.348	0.132	0.221	**0.067**	**0.182**
	672	0.282	0.363	0.259	0.356	0.259	0.359	0.256	0.356	0.148	0.251	**0.070**	**0.184**
	1344	0.336	0.408	0.279	0.404	0.296	0.384	0.279	0.384	0.208	0.252	**0.103**	**0.245**
Solar	96	0.379	0.326	0.369	0.336	0.435	0.414	0.412	0.452	0.353	0.342	**0.281**	**0.295**
	384	0.388	0.341	0.400	0.342	0.479	0.445	0.430	0.470	0.365	0.353	**0.283**	**0.294**
	672	0.409	0.395	0.394	0.382	0.526	0.496	0.454	0.503	0.402	0.393	**0.293**	**0.315**
	1344	0.506	0.489	0.424	0.396	0.596	0.525	0.616	0.653	0.510	**0.381**	**0.365**	0.384
Exchange	96	1.503	1.086	1.395	1.076	0.715	0.625	0.692	**0.608**	0.906	0.736	**0.635**	0.618
	384	1.760	1.117	1.417	1.132	1.276	0.888	0.722	0.682	0.948	0.745	**0.682**	**0.646**
	672	1.695	1.293	1.596	1.325	0.775	0.695	0.815	0.769	1.103	0.964	**0.725**	**0.695**
	1344	1.774	1.924	1.893	1.725	1.102	0.995	1.125	1.084	1.354	1.251	**0.907**	**0.852**
ECL	96	0.476	0.499	0.403	0.451	0.764	0.659	0.478	0.550	0.469	0.486	**0.376**	**0.428**
	384	0.484	0.500	0.425	0.480	0.778	0.681	0.489	0.542	0.493	0.495	**0.367**	**0.417**
	672	0.503	0.503	0.443	0.506	0.826	0.706	0.496	0.562	0.524	0.517	**0.377**	**0.423**
	1344	0.623	0.609	**0.503**	**0.552**	0.923	0.797	0.620	0.789	0.623	0.526	0.712	0.622
Count		1		4		0		1		6		28	

Table 1: Univariate long sequence time-series forecasting results on five datasets.

2016 to July 2, 2018 on different sensors on highways in the San Francisco Bay Area, recorded every hour, with each record containing 862 attributes.

- Exchange (Lai et al. 2018): Records daily exchange rates from January 1, 1990 to October 10, 2010. The dataset has a total of 8 attributes and 7,588 records.
- ECL (Trindade 2015): Records the daily electricity usage of 321 customers from July 1, 2016 to July 2, 2019.
- Solar: This is a dataset that we have built by deploying data collection terminals at photovoltaic power stations in northern China, recording the photovoltaic power generation situation from June 30, 2020 to June 30, 2023. Each record has 8 attributes, including power generation, total radiation, direct radiation, scattered radiation, temperature, air pressure, wind direction, and wind speed. The data is sampled every 15 minutes and there are a total of 105216 records.

The training set, validation set and test set of each dataset are divided according to the ratio of 0.7, 0.1 and 0.2.

Experimental Details. We use L2 loss (MSE) as the loss function during training and Adam optimizer as the optimizer. The initial learning rate is set to 0.0001 and the batch size is set to 32. The basic hyperparameters of the model include the number of encoder layers, the number of decoder layers, model depth, number of attention heads and convolution function dimension, which are set to 2,1,512,8 and 2048 respectively. The window length hyperparameter is set to 'auto'. We repeat each experiment 3 times and the mean of the metrics reported.

For metric, we ues two evalution metrics, including $MSE = \frac{1}{n}\sum_{i=1}^{n}(y - \hat{y})^2$ and $MAE = \frac{1}{n}\sum_{i=1}^{n}|y - \hat{y}|$. For the software and hardware relied on in the experiment, all models were trained and tested on a single Nvidia A100 GPU with 40GB memories, operating system Ubuntu 20.04.3, and model framework PyTorch 11.7.

Benchmark Model. In order to objectively evaluate our method, we used 5 popular transformer-based time series prediction model as the benchmark: Transformer (Wu et al. 2020), Informer (Zhou et al. 2021), Autoformer (Wu et al. 2021), FEDformer (Zhou et al. 2022) and TimesNet (Wu et al. 2023). In order to enhance the competitiveness of benchmark models, their hyperparameters have been adjusted to ensure that they are in their best fit state.

Compare the Experimental Results

Table 1 and Table 2 show the results of univariate-to-univariate and multivariate-to-univariate forecasting on five datasets, respectively. The best results are highlighted in boldface. Their input lengths are all 384, and their prediction lengths are 96, 384, 672, and 1344, respectively, to simulate short-term prediction, medium-term prediction, medium-long-term prediction, and long-term prediction.

Univariate-to-Univariate Forecasting. We conducted an experiment of univariate-to-univariate forecasting on 5 datasets, and the results are shown in Table 1. It can be seen that our proposed ISDformer achieved 28 top-1 and 30 top-2 cases out of 40 in total, significantly better than the comparison methods. Our method achieved almost the best results in the Traffic, Solar, ECL, and Exchange datasets. In particular,

Method		Informer		Transformer		Autoformer		FEDformer		TimesNet		**ISDformer**	
Sources		(Zhou et al. 2021)		(Wu et al. 2020)		(Wu et al. 2021)		(Zhou et al. 2022)		(Wu et al. 2023)		**Ours**	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTm2	96	0.370	0.483	0.352	0.469	0.355	0.468	1.297	0.876	**0.302**	**0.396**	0.585	0.596
	384	0.375	0.488	0.370	0.481	0.385	0.493	1.315	0.900	**0.321**	**0.424**	0.603	0.643
	672	0.406	0.534	0.386	0.519	0.408	0.513	1.426	0.957	**0.347**	**0.436**	0.610	0.604
	1344	0.484	0.550	0.387	0.499	**0.357**	**0.471**	0.957	0.782	0.509	0.545	0.723	0.836
Traffic	96	0.853	0.764	0.776	0.758	0.272	0.306	0.363	0.397	0.132	0.168	**0.107**	**0.136**
	384	0.962	0.749	0.896	0.923	0.288	0.393	0.379	0.408	0.149	0.196	**0.106**	**0.140**
	672	1.126	1.253	1.026	1.089	0.353	0.420	0.390	0.413	0.168	0.253	**0.156**	**0.191**
	1344	1.942	1.108	0.961	0.907	0.283	0.390	0.419	0.434	0.218	0.297	**0.191**	**0.226**
Solar	96	0.488	0.397	0.446	0.353	0.446	0.423	2.375	1.627	0.488	0.316	**0.279**	**0.297**
	384	0.497	0.371	0.476	0.373	0.467	0.444	2.453	1.625	0.497	0.327	**0.286**	**0.308**
	672	0.497	0.375	0.470	0.373	0.594	0.503	2.512	1.630	0.515	**0.336**	**0.325**	0.358
	1344	0.422	0.384	0.460	**0.368**	0.444	0.458	2.586	1.856	0.499	0.384	**0.378**	0.406
Exchange	96	1.595	1.106	1.429	1.098	0.779	0.652	1.594	1.126	1.498	0.885	**0.612**	**0.591**
	384	1.641	1.123	1.492	1.103	0.784	0.678	1.705	1.036	1.507	0.890	**0.594**	**0.578**
	672	1.736	1.243	1.519	1.227	0.796	0.709	1.736	1.206	1.523	0.919	**0.597**	**0.663**
	1344	2.354	2.244	1.936	1.857	1.328	**1.227**	2.193	2.335	1.932	1.846	**1.125**	1.352
ECL	96	0.495	0.592	0.486	**0.424**	0.806	0.907	0.491	0.525	0.479	0.504	**0.443**	0.463
	384	0.593	0.623	0.505	0.556	0.828	0.749	0.519	0.541	**0.490**	0.551	0.527	**0.536**
	672	0.625	0.657	0.560	**0.526**	0.926	0.896	**0.556**	0.574	0.564	0.595	0.607	0.658
	1344	0.927	0.778	**0.350**	**0.447**	0.394	0.464	0.564	0.568	0.635	0.596	0.837	0.857
Count		1		5		3		1		8		23	

Table 2: Multivariate long sequence time-series forecasting results on five datasets.

ISDformer performed extremely well in the traffic flow prediction task, with the maximum MSE decrease of 59%, 80%, 76%, 77%, and 83% compared to Timesnet, FEDformer, Autoformer, Transformer, and Informer, respectively. This shows that the learning paradigm of ISDformer can effectively capture the potential asynchronous characteristics of time series variables, thereby improving the predictive performance of the model.

Multivariate-to-Univariate Forecasting. We conducted an experiment of multivariate-to-univariate forecasting on 5 datasets, and the results are shown in Table 2. It can be seen that ISDformer achieved 23 top-1 and 26 top-2. Similarly, almost the best results were achieved in the Traffic, Solar, ECL, and Exchange datasets. ISDformer's MSE decreased by 60% (Exchange $\tau = 384$), 88% (Solar $\tau = 384$), 63% (Traffic $\tau = 672$), 88% (Traffic $\tau = 672$), and 90% (Traffic $\tau = 1344$) compared to Timesnet, FEDformer, Autoformer, Transformer and Informer, respectively.

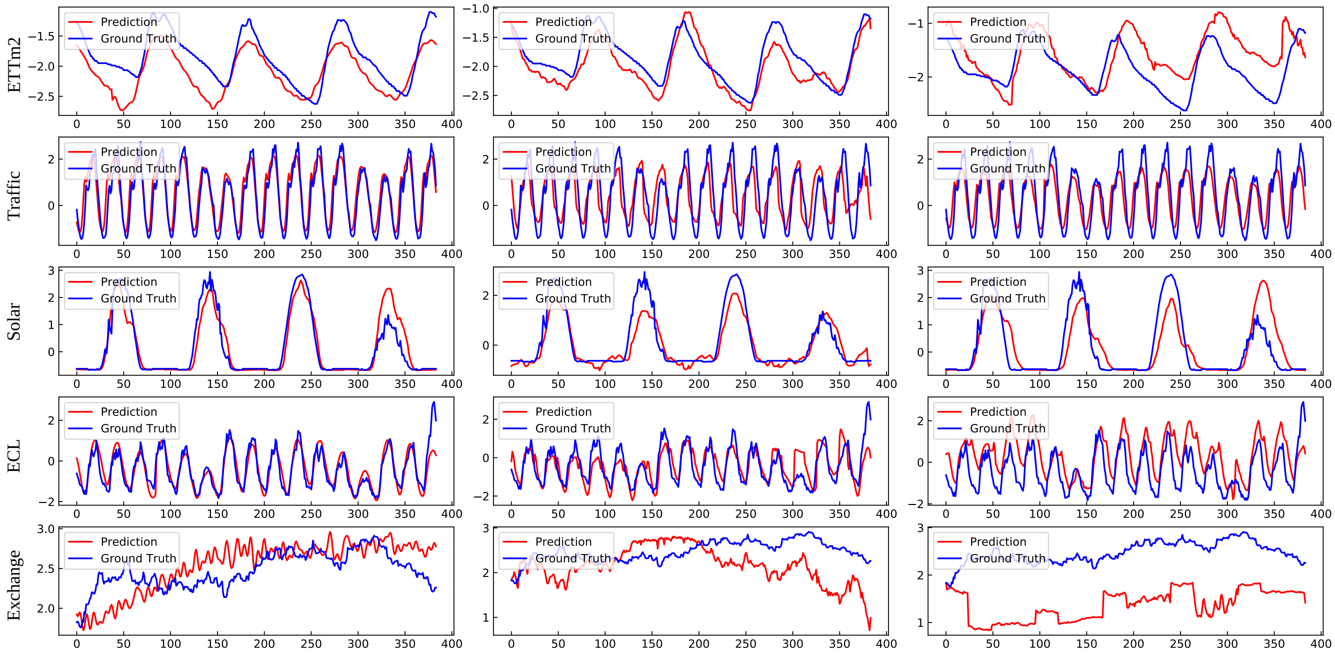
Analyzing table 1 and table 2, we can find that ISDformer's multivariate-to-univariate performance in the ETTm2 and Exchange datasets is better than univariate-to-univariate performance, which shows that ISDformer's decoupled learning paradigm can effectively capture the asynchronous spatiotemporal relationship of time series variables and thereby improve the predictive performance of the model. Compared with other models, under the same task, their multivariate-to-univariate predictive performance is generally worse than univariate-to-univariate predictive performance.

In the Solar dataset, the average MSE of the multivariate-to-univariate task is only 3.6% different from univariate-to-univariate. This is because compared to the Exchange and ETT datasets, the Solar dataset is more regular, and introducing covariates does not help much to improve model performance. In addition, for the Traffic dataset and ECL dataset, in the experiment of multivariate prediction of univariate, its huge covariate scale also caused a decline in the proposed model. This is because in the VEA mechanism, all word vectors share a set of **Q**, **K**, and **V** parameters. The increase in the number of covariates will cause the model fitting performance to deteriorate and ultimately lead to a decline in predictive performance. At the same time, we found that ISDformer's performance on the ETTm2 dataset was lower than that of comparison models. This is because this data has strong local trend characteristics. The introduction of frequency domain assignment processing during encoding weakens the model's learning ability for local trends in time dimension and enhances overall trend characterization.

Figure 5 shows the visualization results of ISDformer, Transformer, and Autoformer models on five datasets in the multivariate to univariate prediction task. It can be seen that compared to other models, the multi periodicity and global trend of time series have been more effectively learned due to the cross dimensional decoupling and frequency domain learning of ISDformer.

Ablation Experiment

In our method, there are 3 key components: F&V embedding, vectors enhancement attention (VEA), and spatiotemporal cross aggregation (SCA). F&V embedding consists of



(a) ISDformer

(b) Autoformer

(c) Transformer

Figure 5: Visualization of prediction results on five datasets (input time-series length $\tau = 384$ and output length $\tau = 384$).

two core points: one is to encode each variable independently in order to capture the potential connections between variables at different times and at the same time; the other is to add frequency domain information under the encoding result. The core idea of VEA is to emphasize the components that are useful for forecasting results; SCA is a feature aggregation mechanism designed to cooperate with independent coding. From an ideological point of view, the three components we propose are trying to solve three problems: the interference problem existing in the data, the problem of capturing asynchronous spatiotemporal relationships, and the problem of how to enhance features.

In order to verify whether the 3 components have better performance on the dataset, we designed the following models for experiments:

- Complete ISDformer (Group A);
- ISDformer without frequency components (Group B);
- Standard Transformer for time series forecasting (Group C);
- Transformer with vectors enhancement attention (VEA) (Group D).

Group	Method	MSE	MAE
A	ISDformer	0.286	0.308
B	ISDformer without frequency	0.404	0.353
C	Transformer	0.476	0.373
D	Transformer with VEA	0.454	0.370

Table 3: Ablation experiment results

All four models are trained on the Solar dataset, and their input and output lengths are both 384. The test results are shown in table 3. It can be seen that in the comparison between group A and group B, group A achieved better results, indicating that frequency encoding has an enhancing effect on the results, that is, it solves the data interference problem mentioned. In the comparison experiment between group C and group D, group D achieved better results, indicating that vectors enhancement attention has good results for feature enhancement. In the comparison experiment between group B and group D, since Transformer with vectors enhancement attention mechanism is equivalent to ISDformer with frequency component removed and coupled encoding, their comparison relationship indicates that the model can capture asynchronous spatiotemporal relationships.

Conclusion and Future Directions

This paper undertakes an exploration of the limitations associated with transformer-based models that inherit NLP paradigms for asynchronous spatiotemporal feature learning in time series data. To address these limitations, a novel decoupled spatiotemporal learning model for time series, named ISDformer, is proposed. Theoretical analysis and empirical experimentation demonstrate a significant enhancement in predictive performance across five challenging datasets through the utilization of the ISDformer framework. Subsequently, our research will shift focus towards the investigation of the aggregative effects of weakly correlated signals during temporal extension, alongside an exploration of the theoretical framework of energy diffusion for medium to long-term prediction tasks.

References

Angryk, R. A.; Martens, P. C.; Aydin, B.; Kempton, D.; Mahajan, S. S.; Basodi, S.; Ahmadzadeh, A.; Cai, X.; Filali Boubrahimi, S.; Hamdi, S. M.; et al. 2020. Multivariate time series dataset for space weather data analytics. *Scientific data*, 7(1): 227.

Barros, J.; Araujo, M.; and Rossetti, R. J. 2015. Short-term real-time traffic prediction methods: A survey. In *2015 International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS)*, 132–139. IEEE.

Biswas, S. K.; Sinha, N.; Purkayastha, B.; and Marbaniang, L. 2014. Weather prediction by recurrent neural network dynamics. *International Journal of Intelligent Engineering Informatics*, 2(2-3): 166–180.

Borovykh, A.; Bohte, S.; and Oosterlee, C. W. 2017. Conditional time series forecasting with convolutional neural networks.

Bouktif, S.; Fiaz, A.; Ouni, A.; and Serhani, M. A. 2020. Multi-sequence LSTM-RNN deep learning and metaheuristics for electric load forecasting. *Energies*, 13(2): 391.

Cheng, Y.-C.; and Li, S.-T. 2011. Fuzzy time series forecasting with a probabilistic smoothing hidden Markov model. *IEEE Transactions on Fuzzy Systems*, 20(2): 291–304.

Demirel, Ö. F.; Zaim, S.; Çalişkan, A.; and Özuyar, P. 2012. Forecasting natural gas consumption in Istanbul using neural networks and multivariate time series methods. *Turkish Journal of Electrical Engineering and Computer Sciences*, 20(5): 695–711.

Ding, X.; Zhang, Y.; Liu, T.; and Duan, J. 2015. Deep learning for event-driven stock prediction. In *Twenty-fourth international joint conference on artificial intelligence*.

Elamin, N.; and Fukushige, M. 2018. Modeling and forecasting hourly electricity demand by SARIMAX with interactions. *Energy*, 165: 257–268.

Galicia, A.; Talavera-Llames, R.; Troncoso, A.; Koprinska, I.; and Martínez-Álvarez, F. 2019. Multi-step forecasting for big data time series based on ensemble learning. *Knowledge-Based Systems*, 163: 830–841.

Hanif, M.; Sami, F.; Hyder, M.; Ch, M. I.; et al. 2017. Hidden Markov model for time series prediction. *Journal of Asian Scientific Research*, 7(5): 196–205.

Ho, S.-L.; Xie, M.; and Goh, T. N. 2002. A comparative study of neural network and Box-Jenkins ARIMA modeling in time series prediction. *Computers & Industrial Engineering*, 42(2-4): 371–375.

Hochreiter, S.; and Schmidhuber, J. 1997. Long short-term memory. *Neural computation*, 9(8): 1735–1780.

Khan, S.; Naseer, M.; Hayat, M.; Zamir, S. W.; Khan, F. S.; and Shah, M. 2022. Transformers in vision: A survey. *ACM computing surveys (CSUR)*, 54(10s): 1–41.

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

Li, J.; Wu, B.; Sun, X.; and Wang, Y. 2021. Causal hidden markov model for time series disease forecasting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 12105–12114.

Li, S.; Jin, X.; Xuan, Y.; Zhou, X.; Chen, W.; Wang, Y.-X.; and Yan, X. 2019. Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting.

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

Ma, X.; Tao, Z.; Wang, Y.; Yu, H.; and Wang, Y. 2015. Long short-term memory neural network for traffic speed prediction using remote microwave sensor data. *Transportation Research Part C: Emerging Technologies*, 54: 187–197.

Patton, A. 2013. Copula methods for forecasting multivariate time series. *Handbook of economic forecasting*, 2: 899–960.

PeMS. 2014. http://pems.dot.ca.gov/.

Salinas, D.; Flunkert, V.; Gasthaus, J.; and Januschowski, T. 2020. DeepAR: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting*, 36(3): 1181–1191.

Sapankevych, N. I.; and Sankar, R. 2009. Time series prediction using support vector machines: a survey. *IEEE computational intelligence magazine*, 4(2): 24–38.

Tarsitano, A.; and Amerise, I. L. 2017. Short-term load forecasting using a two-stage sarimax model. *Energy*, 133: 108–114.

Thissen, U.; Van Brakel, R.; De Weijer, A.; Melssen, W.; and Buydens, L. 2003. Using support vector machines for time series prediction. *Chemometrics and intelligent laboratory systems*, 69(1-2): 35–49.

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

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

Wen, Q.; Zhou, T.; Zhang, C.; Chen, W.; Ma, Z.; Yan, J.; and Sun, L. 2022. Transformers in time series: A survey.

Wu, H.; Hu, T.; Liu, Y.; Zhou, H.; Wang, J.; and Long, M. 2023. TimesNet: Temporal 2D-Variation Modeling for General Time Series Analysis. In *The Eleventh International Conference on Learning Representations,ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

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

Wu, N.; Green, B.; Ben, X.; and O'Banion, S. 2020. Deep transformer models for time series forecasting: The influenza prevalence case. *arXiv preprint arXiv:2001.08317*.

Yang, H.; Pan, Z.; Tao, Q.; and Qiu, J. 2018. Online learning for vector autoregressive moving-average time series prediction. *Neurocomputing*, 315: 9–17.

Zhang, J.; and Man, K.-F. 1998. Time series prediction using RNN in multi-dimension embedding phase space. In *SMC'98 Conference Proceedings. 1998 IEEE International Conference on Systems, Man, and Cybernetics (Cat. No. 98CH36218)*, volume 2, 1868–1873. IEEE.

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

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

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