HDMixer: Hierarchical Dependency with Extendable Patch for Multivariate Time Series Forecasting

Qihe Huang[1,4,†], Lei Shen[4], Ruixin Zhang[4], Jiahuan Cheng[3,4], Shouhong Ding[4], Zhengyang Zhou[1,2,5,*], Yang Wang[1,2,*]

[1] University of Science and Technology of China (USTC), Hefei, China
[2] Suzhou Institute for Advanced Research, USTC, Suzhou, China
[3] Johns Hopkins University
[4] Youtu Laboratory, Tencent, Shanghai, China
[5] State Key Laboratory of Resources and Environmental Information System
hqh@mail.ustc.edu.cn, {zzy0929, angyan}@ustc.edu.cn, {shenlei1996,chengjoanna56}@gmail.com, {ruixinzhang, ericshding}@tencent.com

Abstract

Multivariate time series (MTS) prediction has been widely adopted in various scenarios. Recently, some methods have employed patching to enhance local semantics and improve model performance. However, length-fixed patch are prone to losing temporal boundary information, such as complete peaks and periods. Moreover, existing methods mainly focus on modeling long-term dependencies across patches, while paying little attention to other dimensions (e.g., short-term dependencies within patches and complex interactions among cross-variavle patches). To address these challenges, we propose a pure MLP-based HDMixer, aiming to acquire patches with richer semantic information and efficiently modeling hierarchical interactions. Specifically, we design a Length-Extendable Patcher (LEP) tailored to MTS, which enriches the boundary information of patches and alleviates semantic incoherence in series. Subsequently, we devise a Hierarchical Dependency Explorer (HDE) based on pure MLPs. This explorer effectively models short-term dependencies within patches, long-term dependencies across patches, and complex interactions among variables. Extensive experiments on 9 real-world datasets demonstrate the superiority of our approach. The code is available at https://github.com/hqh0728/HDMixer.

Introduction

Multivariate time series prediction has become an integral component in various real-world applications (Wang et al. 2023f; Xu et al. 2021; Chen et al. 2024; Xu et al. 2020; Zhou et al. 2020a; Zhang et al. 2023; Shen et al. 2022; Fang et al. 2023), such as weather forecasting, power scheduling, and economic assessment (Cheng et al. 2022; Kim et al. 2021; Shang, Chen, and Bi 2021; Tian et al. 2021; Tang, Lu, and Qiu 2023b,a; Yang et al. 2023a). Deep learning has greatly improved the MTS prediction performance (Yin and Shang 2016; Liu et al. 2022, 2021; Zhou et al. 2023a; Wang et al. 2021; Zhou et al. 2020b), leading to a proliferation of deep forecasting models (Wu et al. 2021, 2019; Shabani et al. 2023; Miao et al. 2024, 2023; Huang et al. 2023). From a technical perspective, extending one-dimensional time series to high-dimensional sequences through temporal patching can enhance the local semantics at each time step and provides significant advantages for subsequent dependency structure mining (Nie et al. 2023; Zhang and Yan 2023). The outstanding performance attained by advanced deep prediction methods is attributed to their initial application of sequence patching, followed by the learning of long-term dependencies (Lin et al. 2023a,b).

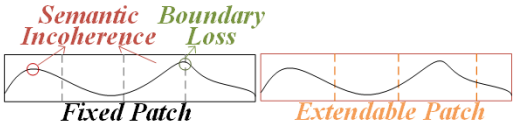

Figure 1: Comparison between Fixed Patch and Extendable Patch.

Generally, these approaches (Nie et al. 2023) use length-fixed patches and make the intuitive assumption that a fixed partitioning of time series applies universally. However, such fixed patches lack flexibility in dividing time series into subsequences, leading to two issues: (1) Loss of boundary information in patch. Subsequences lose crucial boundary information, failing to form locally semantic representations with highly informative content. As shown in Fig.1, peak information is lost due to length-fixed patches. (2) Semantic incoherence in sequence. This results in inconsistent handling of semantics across the entire sequence. As shown in Fig.1, peak information inconsistently appears across patches. Consequently, this fixed-patch approach could potentially undermine the semantic information of sequences, leading to a performance decline.

Most existing methods (Nie et al. 2023; Zhang and Yan 2023; Zhang et al. 2022a) focus on enriching local semantics in MTS through patch-based approaches, primarily emphasizing on capturing long-term interactions. However, they tend to overlook two other critical forms of information exchange:

† Work is done when Qihe Huang was an intern at YouTu Lab.
∗ Yang Wang and Zhengyang Zhou are corresponding authors.

(a) short-term interactions and (b) cross-variable interactions. In one hand, capturing short-term interactions facilitates frequent information exchange among neighboring time steps, aiding in identifying local patterns and transient events. In the other hand, extracting cross-variable interactions offers insights into latent causal connections hidden in data variations. Although a few cross-variable techniques (Zhang and Yan 2023) attempt to capture these interactions explicitly, they often involve high computational complexity. Furthermore, internal short-term dependencies within patches have also been overlooked in current research. Integrating interactions across all three dimensions holds the potential to enhance the representation capabilities of models for MTS forecasting.

Based on the aforementioned motivations, we propose HDMixer, a pure MLP-based model to obtain patches with more comprehensive semantic information and model interactions across different dimensions. Specifically, we introduce the Length-Extendable Patcher (LEP) tailored for temporal data, enabling adaptive extension of patch lengths based on temporal characteristics. We innovatively introduce a patch entropy loss for guidance to ensure that the patch division by LEP. For capturing interactions within hierarchical dimensions, we devise a Hierarchical Dependency Explorer (HDE) composed solely of MLPs. It consists of multiple stacked Mixers, each incorporating three MLPs to capture long-term dependencies, short-term dependencies, and cross-variable interactions. Based on pure MLPs, HDE can maintain the computational efficiency and ease of implementation. The main contributions of this paper are summarized as follows:

- We propose a Length-Extendable Patcher (LEP) tailored for temporal data to adaptively enrich the boundary semantics of time series patches. This strategy prevents the loss of complete peak or period. Additionally, we introduce Patch Entropy Loss as an innovative guidance for patch partitioning.
- We devise a Hierarchical Dependency Explorer (HDE) composed solely of MLPs to efficiently model short-term dependencies within patches, long-term inter-patch dependencies, and intricate cross-variable relationships.
- Extensive experimental results on 9 real-world MTS datasets show that our model demonstrates competitive performance. Specifically, it attains top-1 performance in 59 settings and top-2 in 13 settings while maintaining remarkable efficiency.

Related Work

MTS Forecasting with Patch

Recently, patch-based approaches (Nie et al. 2023) for time series prediction have gained significant attention. These models exhibit notable performance improvements compared to methods without patching, underscoring the importance of enhancing local semantics through patching. For instance, Crossformer (Zhang and Yan 2023) segments each time series into patches and employs self-attention mechanisms to model dependencies across variables and time dimensions, leading to considerable performance gains over other methods. Notably, PatchTST (Nie et al. 2023) achieves state-of-the-art results by utilizing series patching and instance normalization within the Transformer architecture. However, its computational cost can become prohibitive when dealing with a large number of series variables. In light of promising results in PatchTST, recent works such as Cross-LKTCN (Luo and Wang 2023) and DCdetector (Yang et al. 2023b), have also embraced patch-based partitioning of time series. They leverage patching to enhance sequence representation and improve downstream task performance. Nevertheless, all current methods overlook issues related to fixed patch lengths, which can lead to the loss of boundary information and semantic incoherence within time series data. In this paper, we introduce the Length-Extendable Patcher (LEP) for time series data, which extends patch lengths to enhance the availability of local semantics within segmented sequences.

Hierarchical Interaction in MTS

Introducing hierarchical interactions in multivariate time series prediction allows for the transmission of information across different levels (Hewamalage, Bergmeir, and Bandara 2021; Zhou et al. 2021). This leads to more accurate and comprehensive forecasting results that can better capture the associations and influences among different variables in the real world. Previous research, including Autoformer (Wu et al. 2021), FEDformer (Zhou et al. 2022), ETSformer (Woo et al. 2022), and PatchTST (Nie et al. 2023), focuse solely on long-term dependencies while overlooking variable relationships. Some GNN-based time series prediction methods (Wang et al. 2023e; Wu et al. 2020; Yu, Yin, and Zhu 2018; Zhou et al. 2023b; Yang et al. 2023a; Wang et al. 2023a,c; Zhang et al. 2022b; Wang et al. 2023b) capture variable relationships through fixed graph structures to enhance prediction accuracy. Crossformer (Zhang and Yan 2023) introduced a module for variable relationships and leveraged routing vectors to partially reduce the computational complexity of modeling these relationships. Subsequently, methods like DSformer (Yu et al. 2023) begin to consider variable relationships in MTS prediction. However, these methods tend to be complex, incurring substantial computational costs. Moreover, aside from variable dimensions, current approaches overlook the new dimensions introduced by patching sequences, i.e., short-term interactions within patches, which aid in identifying local patterns and instantaneous events. Motivated by multi-level interactions and computational efficiency (Tolstikhin et al. 2021), we employ a simple yet computationally efficient Mixer to individually address short-term dependencies within patches, long-term dependencies among patches, and variable associations, which promotes an efficient hierarchical interaction scheme.

Methodology

The objective of multivariate time series forecasting is to predict the future values of each variable over T time steps based on their respective historical values over the past L time steps. Given the input matrix $X_{\text{input}} = [X^1, X^2, \ldots, X^M] \in \mathbb{R}^{M \times L}$, where $X^m = [x_1^m, x_2^m, ..., x_L^m] \in R^L$ represents the historical time series of the m-th variable with length L. We aim to learn a function $F : X_{\text{input}} \rightarrow \hat{X}$, where $\hat{X} = [\hat{X}^1, \hat{X}^2, \ldots, \hat{X}^M] \in \mathbb{R}^{M \times T}$, and $\hat{X}^m =$

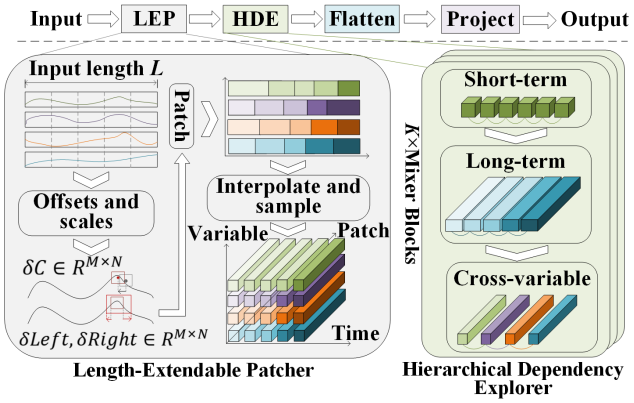

Figure 2: Overall structure of the proposed HDMixer.

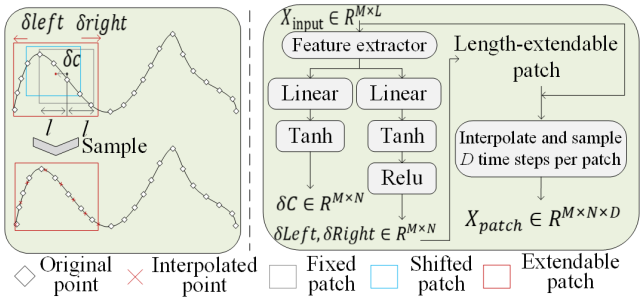

Figure 3: Sturcture of Length-Extendable Patcher(LEP). The gray box is first shifted to become a green box, then undergoes dilation to transform into a red box. Finally, D points are sampled within the red box.

$[\hat{x}^m_{L+1}, \hat{x}^m_{L+2}, \ldots, \hat{x}^m_{L+T}] \in \mathbb{R}^T$ represents the future predicted values of the m-th variable with length T. We propose a pure MLP model called HDMixer, which enables learning of length-extendable patches and interactions among variables in multiple dimensions for time series forecasting. In the following, we will introduce the overall structure of the proposed HDMixer in detail. Then, we present the Length-Extendable Patcher (LEP), which enhances local semantic information through adaptive patch lengths. Finally, the MLP-based Hierarchical Dependency Explorer (HDE) is introduced to facilitate efficient multi-dimensional interactions.

Overall Structure

Forward process The overall architecture of HDMixer is illustrated in Fig.2. We represent the time series of M variables with a historical length of L as $X_{\text{input}} \in \mathbb{R}^{M \times L}$. It is then divided into N length-extendable patches, where each patch contains D evenly sampled values. The process of generating extendable patches for multivariate time series data is described as follows:

$$X_{\text{patch}} = \text{LEP}(X_{\text{input}}) \tag{1}$$

where $LEP(\cdot)$ is the Length-Extendable Patcher function designed specifically for time series data, and its details will be introduced in Section LFP. After applying the extendable patch operation to the multivariate time series, we obtain $X_{\text{patch}} \in \mathbb{R}^{M \times N \times D}$, where N represents the number of patches and D denotes the number of sampled time series values within each patch.

The patched X_{patch} is then fed into the Hierarchical Dependency Explorer to capture long-term dependencies, short-term dependencies, and inter-variable relationships. Thus, based on HDE, we obtain a multivariate time series representation capturing multidimensional dependencies, denoted as $Z^K \in \mathbb{R}^{M \times N \times D}$:

$$Z^K = \text{HDE}(X_{\text{patch}}) \tag{2}$$

where $\text{HDE}(\cdot)$ is the stack of K HDMixer blocks, K indicates the number of blocks, and its detailed description will be presented in Section Hierarchical Dependency Explorer.

Finally, we unroll the last two dimensions of the obtained representation Z^K and apply a mapping function to obtain the final prediction:

$$\hat{X} = Pred(Flatten(Z^K)) \tag{3}$$

where $Flatten(\cdot)$ is used to unroll the last two dimensions of Z, transforming the shape of the final time series representation to $Z^K \in \mathbb{R}^{M \times (D \times N)}$. $Pred(\cdot)$ maps the final representation to the desired prediction sequence length, and $\hat{X} \in \mathbb{R}^{M \times T}$ represents the final prediction results with length T. M denoting the number of variables. Here, we adopt the design from (Zeng et al. 2022), which employs Directly Multi-Step (DMS) forecasting. This approach directly predicts all time steps in one step, avoiding additional bias caused by error accumulation.

Length-Extendable Patcher

To obtain patches with more comprehensive semantic information, we adaptively expand the patch lengths and employ bi-linear interpolation to uniformly sample a fixed number of D time steps within each patch, resulting in more complete semantic patch partitioning strategy.

Let's consider patching a single-variable time series $X^m = [x^m_1, x^m_2, ..., x^m_T]$ (for simplicity, we omit the subscript m in the following). In the commonly used Length-fixed patch method, p_i can be represented as a tuple (c_i, D), where c_i is the central point coordinate of the i-th patch box, and D represents the number of sampling points. Since the length of the patch in Length-fixed patch equals $D-1$, we can define $l = \frac{D-1}{2}$ as half of the fixed patch length. Consequently, the start and end points of the i-th patch sequence can be calculated as $left_i = c_i - l$ and $right_i = c_i + l$, respectively. Thus, the i-th patch samples the sequence $[x_{c_i-l}, ..., x_{c_i+l}]$ with a sampling interval of 1, obtaining D original observation values from X.

In Length-Extendable Patcher, p_i is expanded into a quintuple $(c_i, D, \delta c_i, \delta left_i, \delta right_i)$, where c_i denotes the central point coordinate of the i-th patch, D is the number of sampling points, δc_i represents the central point offset, $\delta left_i$ represents the left part expansion of the central point, and $\delta right_i$ represents the right part expansion of the central

point. All these δc_i, $\delta left_i$, and $\delta right_i$ are learnable values. Given $l = \frac{D-1}{2}$, the sampling start and end points of the i-th extendable patch are determined as follows:

$$left_i = (c_i + \delta c_i) - (l + \delta left_i) \tag{4}$$

$$right_i = (c_i + \delta c_i) + (l + \delta right_i) \tag{5}$$

The corresponding sampling sequence is $[x_{left_i}, ..., x_{right_i}]$, with a sampling interval of $\frac{2l+\delta left_i+\delta right_i}{D-1}$ and a total of D values. To learn the central point offset δC, left and right boundary expansion lengths δL, δR for all patches of each variable sequence, we design a branch as follows:

$$\delta C = \frac{D}{a} \cdot \text{Tanh}(W_{offset} \cdot f(X_{input})) \tag{6}$$

$$\delta L, \delta R = \frac{D}{b} \cdot \text{ReLU}\left(\text{Tanh}(W_{extend} \cdot f(X_{input}))\right) \tag{7}$$

Here, $f(\cdot)$ serves as a feature extractor, designed as a single-layer linear function. $W_{offset} \in \mathbb{R}^{T\times N}$ and $W_{extend} \in \mathbb{R}^{T\times 2N}$ are used to learn the offset and length scales of the patch. $\delta C \in \mathbb{R}^{M\times N}$ represents the central point offset for all patches of the multivariate variables, ensuring that the central point offset falls within $[-\frac{D}{a}, \frac{D}{a}]$ using $\frac{D}{a}$. The hyperbolic tangent function $\tanh(\cdot)$. δL and $\delta R \in \mathbb{R}^{M\times N}$ represent the left and right boundary expansion lengths of all patches for each variable, respectively. The use of ReLU function and $\frac{D}{b}$ ensures that the boundary expansion lengths are within the range $[0, \frac{D}{b}]$. By limiting the patch offset and expandable length through two coefficients, the overlap between patches is reduced, avoiding excessive similarity between adjacent patches.

Based on the learnable offset δC and length expansion $\delta L, \delta R$, we determine the start and end points of each patch and sample the time series accordingly. For Length-Extendable Patch, the main challenge is that the index of the uniformly sampled sequence is generally fractional. To address this, we use bi-linear interpolation to obtain the corresponding sampling points, while ensuring differentiability. For a fractional index $i = left + k \cdot \frac{2l+\delta left+\delta right}{D-1}$, where $k \in [0, D-1]$, we use bi-linear interpolation to obtain its value:

$$X_i = \sum_j G(i; j) \cdot X_j \tag{8}$$

$$G(i; j) = \max(0, 1 - |i - j|) \tag{9}$$

Here, X_i represents the sampled value at the fractional time step, and j iterates over all time steps of the time series. X_j denotes the value of the j-th time step. The function $G(i; j)$ takes a positive value for neighboring integer indices i and j, while zero for other non-adjacent points. We sample the value at a fractional index i based on the surrounding integer indices j through bi-linear interpolation. This ensures the differentiability of the obtained sampling point value with respect to the fractional index i and maintains the overall differentiability of the algorithm. It can be observed that when $i = j$, the sampled value corresponds to the initial sequence value X_j. Using the above equation, we can efficiently compute the sampling points for all length-extendable patches of each variable, requiring only a small number of multiply-add operations. The final output of the Length-Extendable Patcher is the three-dimensional vector $X_{patch} \in \mathbb{R}^{M\times N\times D}$ obtained from the two-dimensional input data X_{input} after length-extendable patching, where N is the number of patches, and D is the number of sampling points within each patch.

Patch Entropy Loss

To ensure that the extension of patches contributes to information gain, we introduce a strategy based on approximate entropy (Pincus 1991) for measuring the differences between patch partition methods. This strategy reflects the likelihood of discovering new information when changing the patching approach, and a corresponding loss function is designed. We evaluate the complexity between length-extendable and length-fixed patches for the same sequence. Generally, the greater the complexity of subsequence partitioning, the more information is obtained, which benefits prediction tasks (Tolstikhin et al. 2021). For any time series $X \in R^L$, we consider length-fixed patching, where N length-fixed patches of length D are extracted to form $X^{LF} = [[x_1^{LF}, ..., x_D^{LF}], ..., [x_{L-D+1}^{LF}, ..., x_L^{LF}]] \in R^{N\times D}$. Similarly, N length-extendable patches are obtained to form $X^{LE} \in R^{N\times D}$. We evaluate the complexity of the partitioning for X^{LF}.

We measure the similarity of different subpatterns after applying length-fixed patching using the following steps. We compute the distance between each pair of subpatterns $X^{LF}(n)$ and $X^{LF}(m)$ as the maximum absolute difference of their corresponding elements:

$$d(X^{LF}(n), X^{LF}(m)) = max|X^{LF}(n) - X^{LF}(m)| \tag{10}$$

Given a threshold r, we count the number of subpatterns $X^{LF}(m)$ whose distance to other subpatterns is less than r, and calculate the ratio relative to N. This provides an approximate measure of the similarity of the m-th subpattern within the entire sequence X^{LF}:

$$C_m^{LF}(X^{LF}|r) = \frac{1}{N} num\{d(X^{LF}(n), X^{LF}(m)) < r\} \tag{11}$$

To assess the complexity of a full sequence after length-fixed patching, we take the logarithm of $C_n^{LF}(X^{LF}|r)$ for all $n \in [1, N]$, and calculate the average:

$$\Phi^{LF}(X^{LF}|r) = \frac{1}{N}\sum_{n=1}^{N} \ln C_n^{LF}(X^{LF}|r) \tag{12}$$

$\Phi_m^{LF}(X^{LF}|r)$ reflects the similarity of different subpatterns after length-fixed patching. We aim for lower $\Phi_m^{LF}(X^{LF}|r)$ values after patching to enhance the distinctness of semantic information among patches, facilitating interactions and prediction tasks.

Similarly, we compute the similarity of subpatterns after applying length-extendable patching:

$$C_m^{LE}(X^{LE}|r) = \frac{1}{N} num\{d(X^{LE}(n), X^{LE}(m)) < r\} \tag{13}$$

$$\Phi^{LE}(X^{LE}|r) = \frac{1}{N}\sum_{n=1}^{N} \ln C_n^{LE}(X^{LF}|r) \tag{14}$$

The patch entropy of a sequence is defined as:

$$\begin{aligned}
&\mathrm{PaEn}(X^{LE}|r, N) \\
&= \lim_{L\to\infty} \left[\Phi^{LF}(X^{LF}|r) - \Phi^{LE}(X^{LE}|r)\right] \\
&= \lim_{L\to\infty} \frac{1}{N}\left[\sum_{i=1}^{N} \ln C_i^{LF}(X^{LF}|r) - \sum_{i=1}^{N} \ln C_i^{LE}(X^{LE}|r)\right] \\
&= \lim_{L\to\infty} 1N \sum_{i=1}^{N} \ln \frac{C_i^{LF}(X^{LF}|r)}{C_i^{LE}(X^{LE}|r)}
\end{aligned} \tag{15}$$

PaEn indicates the differences in similarity between subpatterns obtained through length-extendable and length-fixed patching. A higher PaEn implies lower similarity among subpatterns after length-extendable patching (indicating higher partitioning complexity and more information), while a lower PaEn implies higher similarity (indicating lower partitioning complexity and less information). Thus, PaEn theoretically characterizes the irregularity and complexity of signal partitioning. It roughly corresponds to the average logarithmic conditional probability of new patterns appearing when patches change. Based on this, we design the information gain loss for length-extendable patches:

$$L_p = -(\Phi^{LE}(X^{LE}|r) - \Phi^{LF}(X^{LF}|r)) \tag{16}$$

This loss minimizes the patch entropy of length-extendable patches, increasing the complexity of obtained patch series to acquire more information suitable for MTS.

Hierarchical Dependency Explorer

The input to HDE is a three-dimensional vector, $X_{patch} \in \mathbb{R}^{M\times N\times D}$, representing the variable dimension, time dimension, and patch internal dimension, respectively. HDE is formed by stacking several HDMixer blocks, and we denote Z^{k+1} as the output of the k-th HDMixer block in HDE, with $Z^k \in \mathbb{R}^{M\times N\times D}$, $k \in \{1, ..., K\}$ representing the input of the k-th HDMixer block, and Z^1 is initialized as X_{patch}. Specifically, an HDMixer block contains three layers of MLPs to capture dependencies in different dimensions. The detailed forward process of the k-th HDMixer block is as follows:

$$U_{*,*,d}^k = Z_{*,*,d}^k + W_{st2} \cdot \mathrm{GELU}(W_{st1} \cdot \mathrm{LN}(Z^k)_{*,*,d}) \tag{17}$$

$$V_{*,n,*}^k = Z_{*,n,*}^k + W_{lt2} \cdot \mathrm{GELU}(W_{lt1} \cdot \mathrm{LN}(U^k)_{*,n,*}) \tag{18}$$

$$Y_{m,*,*}^k = Z_{m,*,*}^k + W_{v2} \cdot \mathrm{GELU}(W_{v1} \cdot \mathrm{LN}(V^k)_{m,*,*}) \tag{19}$$

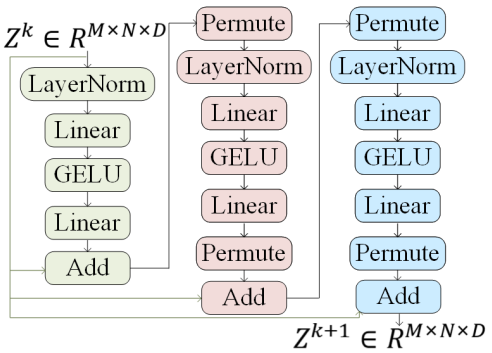

Figure 4: Structure of Hierarchical Dependency Explorer(HDE). The sequence undergoes short-term interactions(green) followed by long-term relationship capturing(red), and finally variable dependency mining(blue).

$$Z^{k+1} = Y^k \tag{20}$$

Here, $\mathrm{GELU}(\cdot)$ is the element-wise activation function, and $\mathrm{LayerNorm}(\cdot)$ is the layer normalization specific to certain dimensions. $U^k \in \mathbb{R}^{M\times N\times D}$ captures short-term dependencies within patches through the channel-mixing MLP, with $W_{st1} \in \mathbb{R}^{M\times 2M}$ and $W_{st2} \in \mathbb{R}^{2M\times M}$ being the learnable matrices corresponding to the two layers of the MLP. Similarly, $V^k \in \mathbb{R}^{M\times N\times D}$ captures long-term dependencies between patches through the time-mixing MLP, with $W_{lt1} \in \mathbb{R}^{N\times 2N}$ and $W_{lt2} \in \mathbb{R}^{2N\times N}$ being the learnable matrices for the two layers of the MLP. $Y^k \in \mathbb{R}^{M\times N\times D}$ captures cross-variable dependencies through the variable-mixing MLP, with $W_{v1} \in \mathbb{R}^{\times 2M}$ and $W_{v2} \in \mathbb{R}^{2M\times M}$ being the learnable matrices for the two layers of the MLP. The final output $Z^{k+1} \in \mathbb{R}^{M\times N\times D}$ is obtained based on the residual connection (He et al. 2015).

It is worth noting that each layer of HDMixer has the same input dimension, unlike the pyramid architecture in CNN, where deeper layers have lower resolutions. This design ensures that effective information from long time series is preserved. Additionally, in our HDMixer, when performing interaction operations in one dimension, the parameters are shared across other dimensions. This approach has two advantages: 1) Significant memory saving: The network's size does not grow rapidly with an increase in the number of variables or sequence length. 2) Enhanced generalization and robustness: Since the time series patterns of different variables are similar, learning their common characteristics can improve the model's generalization performance.

In conclusion, the output of the k-th HDMixer block can be simplified as:

$$Z^{k+1} = \mathrm{HDMixer}^k(Z^k) + Z^k \tag{21}$$

Here, $\mathrm{HDMixer}^k(\cdot)$ represents the k-th HDMixer block, corresponding to equations (17), (18), and (19), which respectively capture variable relationships, long-term dependencies, and short-term dependencies. The final output of HDE is $Z^K \in \mathbb{R}^{M\times N\times D}$, where K denotes the number of HDMixer blocks in HDE.

Models		HDMixer		PatchTST		DLinear		Crossformer		MICN		Autoformer		MTGNN	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ILI	24	**1.305**	**0.767**	1.522	0.814	2.215	1.081	3.040	1.186	2.684	1.112	2.906	1.182	4.268	1.385
	36	**1.428**	**0.763**	1.430	0.834	1.963	0.963	3.356	1.230	2.507	1.013	2.585	1.038	4.768	1.494
	48	**1.233**	**0.798**	1.673	0.854	2.130	1.024	3.441	1.223	2.423	1.012	3.024	1.145	5.333	1.592
	60	**1.496**	**0.853**	1.529	0.862	2.368	1.096	3.608	1.302	2.653	1.085	2.761	1.114	5.083	1.556
Weather	96	**0.145**	**0.199**	0.152	0.199	0.176	0.237	0.153	0.217	0.161	0.226	0.249	0.329	0.161	0.223
	192	**0.194**	**0.241**	0.197	0.243	0.220	0.282	0.197	0.269	0.220	0.283	0.325	0.370	0.206	0.278
	336	**0.237**	**0.281**	0.249	0.283	0.265	0.319	0.252	0.311	0.275	0.328	0.351	0.391	0.261	0.322
	720	**0.317**	**0.333**	0.320	0.335	0.323	0.362	0.318	0.363	0.323	0.356	0.415	0.426	0.324	0.366
Traffic	96	**0.366**	**0.249**	0.367	0.251	0.410	0.282	0.512	0.290	0.508	0.301	0.597	0.371	0.527	0.316
	192	**0.385**	**0.258**	**0.385**	0.259	0.423	0.287	0.523	0.297	0.536	0.315	0.607	0.382	0.534	0.320
	336	0.403	0.276	**0.398**	**0.265**	0.436	0.296	0.530	0.300	0.525	0.310	0.623	0.387	0.540	0.335
	720	0.441	0.295	**0.434**	**0.287**	0.466	0.315	0.573	0.313	0.571	0.323	0.639	0.395	0.557	0.343
Electricity	96	**0.129**	**0.219**	0.130	0.222	0.140	0.237	0.187	0.283	0.159	0.267	0.196	0.313	0.198	0.294
	192	**0.145**	**0.235**	0.148	0.240	0.153	0.249	0.258	0.330	0.168	0.279	0.211	0.324	0.266	0.339
	336	0.170	**0.259**	**0.167**	0.261	0.169	0.267	0.323	0.369	0.196	0.308	0.214	0.327	0.328	0.373
	720	**0.193**	**0.289**	0.202	0.291	0.203	0.301	0.404	0.423	0.203	0.312	0.236	0.342	0.422	0.410
ETTh1	96	**0.373**	**0.398**	0.375	0.399	0.375	0.399	0.386	0.429	0.396	0.427	0.435	0.446	0.439	0.461
	192	0.412	0.420	0.414	0.421	**0.405**	**0.416**	0.419	0.444	0.430	0.453	0.456	0.457	0.476	0.477
	336	**0.392**	**0.417**	0.431	0.436	0.439	0.443	0.440	0.461	0.433	0.458	0.486	0.487	0.736	0.643
	720	**0.448**	**0.463**	0.449	0.466	0.472	0.490	0.519	0.524	0.474	0.508	0.515	0.517	0.916	0.750
ETTh2	96	**0.262**	**0.329**	0.274	0.336	0.289	0.353	0.628	0.563	0.289	0.357	0.332	0.368	0.690	0.614
	192	**0.317**	**0.367**	0.339	0.379	0.383	0.418	0.703	0.624	0.409	0.438	0.426	0.434	0.745	0.662
	336	**0.308**	**0.367**	0.331	0.380	0.448	0.465	0.827	0.675	0.417	0.452	0.477	0.479	0.886	0.721
	720	0.390	**0.421**	**0.379**	0.422	0.605	0.551	1.181	0.840	0.426	0.473	0.453	0.490	1.299	0.936
ETTm1	96	0.291	**0.341**	**0.290**	0.342	0.299	0.343	0.316	0.373	0.314	0.360	0.510	0.492	0.428	0.446
	192	**0.332**	**0.364**	**0.332**	0.369	0.335	0.365	0.377	0.411	0.359	0.387	0.514	0.495	0.509	0.491
	336	**0.363**	**0.385**	0.366	0.392	0.369	0.386	0.431	0.442	0.398	0.413	0.510	0.492	0.577	0.556
	720	0.424	**0.417**	**0.420**	0.424	0.425	0.421	0.600	0.547	0.459	0.464	0.527	0.493	0.713	0.729
ETTm2	96	**0.162**	**0.254**	0.165	0.255	0.167	0.260	0.421	0.461	0.178	0.273	0.205	0.293	0.463	0.503
	192	**0.213**	**0.289**	0.220	0.292	0.224	0.303	0.503	0.519	0.245	0.316	0.278	0.336	0.530	0.547
	336	**0.275**	0.331	0.278	**0.329**	0.281	0.342	0.611	0.580	0.295	0.350	0.343	0.379	0.449	0.473
	720	**0.355**	**0.380**	0.367	0.385	0.397	0.421	0.996	0.750	0.389	0.406	0.414	0.419	1.093	0.836
Exchange	96	**0.078**	**0.198**	0.093	0.214	0.081	0.203	0.186	0.346	0.102	0.235	0.197	0.323	0.203	0.381
	192	0.168	**0.289**	0.192	0.312	**0.157**	0.293	0.467	0.522	0.172	0.316	0.300	0.369	0.459	0.512
	336	0.311	**0.396**	0.350	0.432	**0.305**	0.414	0.783	0.721	0.272	0.407	0.509	0.524	0.707	0.697
	720	**0.641**	**0.599**	0.911	0.716	0.643	0.601	1.367	0.943	0.714	0.658	1.447	0.941	1.323	0.912

Table 1: MTS forecasting results in terms of MSE and MAE, the lower the better. For ILI, the prediction length $T \in \{24, 36, 48, 60\}$ and input length is set as 60. As to the other datasets, the prediction length $T \in \{96, 192, 336, 720\}$ and look back window size is set as 336. The best results are highlighted in bold and the second best are underlined.

Experiment

Datasets and Experiment Setup

Datasets We conduct extensive experiments on nine real-world datasets, as outlined in (Wu et al. 2021). These datasets include **Weather**, **Traffic**, **Exchange Rate**, **Electricity**, **ILI**, and four **ETT** datasets (ETTh1, ETTh2, ETTm1, and ETTm2).

Baselines and Setup We compare our method with 6 state-of-the-art methods, including 3 Transfomer-based models: **PatchTST** (Nie et al. 2023), **Crossformer** (Zhang and Yan 2023), **Autoformer** (Wu et al. 2021); Linear-based model: **DLinear** (Zeng et al. 2022); CNN-based model: **MICN** (Wang et al. 2023d); GNN-based method: **MTGNN** (Wu et al. 2020). All the models are following the same experimental setup with prediction length $T \in \{24, 36, 48, 60\}$ for ILI and $T \in \{96, 192, 336, 720\}$ for others. We collect part of baseline results from (Nie et al. 2023). We perform grid search on batch size and learning rate to achieve optimal performance of HDMixer.

Main Results

The quantitative results for MTS forecasting with various baselines are presented in Table 1. HDMixer demonstrates exceptional performance across multiple datasets and prediction length settings, securing 59 first-place and 13 second-place

Dataset		ETTm2				Weather				ILI			
Prediction Length		96	192	336	720	96	192	336	720	24	36	48	60
0.25	MSE	0.169	0.215	0.278	0.360	0.150	0.194	0.244	0.325	1.312	1.432	1.235	1.528
	MAE	0.258	0.294	0.341	0.384	0.206	0.248	0.287	0.334	0.771	0.764	0.799	0.857
0.5	MSE	**0.162**	0.213	**0.274**	**0.355**	0.145	**0.194**	**0.237**	**0.317**	**1.305**	1.428	**1.233**	1.526
	MAE	**0.254**	**0.289**	**0.333**	**0.380**	0.199	**0.241**	**0.281**	**0.333**	**0.767**	**0.763**	**0.798**	0.853
1	MSE	0.164	**0.212**	**0.274**	0.357	**0.144**	0.196	0.239	0.318	1.306	**1.427**	1.234	**1.512**
	MAE	0.256	**0.289**	0.334	**0.380**	**0.198**	0.243	0.282	0.334	0.769	0.764	**0.798**	**0.842**

Table 2: Quantitative results of different extendable length.

Dataset		ETTm2				Weather				ILI			
Prediction Length		96	192	336	720	96	192	336	720	24	36	48	60
HDMixer	MSE	**0.162**	**0.213**	**0.274**	**0.355**	**0.145**	**0.194**	**0.237**	**0.317**	**1.305**	**1.428**	**1.233**	**1.526**
	MAE	**0.254**	**0.289**	**0.333**	**0.380**	**0.199**	**0.241**	**0.281**	**0.333**	**0.767**	**0.763**	**0.798**	**0.853**
W/O-LTD	MSE	0.165	0.222	0.284	0.359	0.153	0.204	0.241	0.322	1.418	1.604	1.304	1.565
	MAE	0.256	0.296	0.340	0.357	0.208	0.255	0.286	0.341	0.793	0.823	0.823	0.871
W/O-VI	MSE	0.169	0.224	0.285	0.365	0.151	0.197	0.237	0.322	1.315	1.582	1.376	1.502
	MAE	0.259	0.295	0.334	0.385	0.200	0.243	0.282	0.339	0.776	0.819	0.825	0.868
W/O-STD	MSE	0.164	0.222	0.276	0.365	0.146	0.195	0.237	0.317	1.561	1.613	1.423	1.596
	MAE	0.254	0.296	0.328	0.385	0.200	0.245	0.285	0.334	0.811	0.841	0.872	0.883
W/O-LEP	MSE	0.169	0.227	0.289	0.369	0.154	0.206	0.249	0.329	1.615	1.625	1.474	1.631
	MAE	0.258	0.299	0.337	0.389	0.209	0.254	0.296	0.346	0.802	0.987	0.888	0.894

Table 3: Ablation of different components in our method.

rankings out of a total of 72 settings. In terms of quantitative measures, HDMixer outperforms Transformer-based methods by achieving an overall 7.27% reduction in MSE and a 4.21% reduction in MAE. Notably, when compared to the GNN-based method MTGNN, HDMixer achieves a 59.66% reduction in MSE and a 42.39% reduction in MAE.

Ablation Study

Study on designed components. We perform ablation studies on three datasets by removing corresponding modules from HDMixer. **W/O-LTD:** Long-term dependency capturing MLP removed. **W/O-STD:** Short-term dependency capturing MLP removed. **W/O-VI:** Variable interaction capturing MLP removed. **W/O-LEP:** Length-Extendable Patcher removed, using fixed patch size of 16. From the results in Table 3, we observe : 1) The predictive performance of W/O-LEP significantly declines, with an average decrease of 5.6% in MSE across 3 datasets, indicating substantial gains from the LEP. 2) Across different datasets, we notice varying benefits from interactions in different dimensions. For instance, the ETTm2 relies more on capturing variable interactions, while the Weather emphasizes long-term dependency.

Study on hyper parameter sensitivity. We investigate the impact of different extension lengths on Table 2. We vary the maximum extended patch lengths to be 0.25, 0.5 and 1 times the original length, respectively. We observe that smaller extension lengths lead to modest improvements in prediction performance, while extensions beyond 0.5 times notably enhance the performance. Optimal improvements are achieved when patch maximum extension lengths are set at 0.5 times and 1 times. Considering overall prediction performance and the need to avoid excessive semantic overlap, we ultimately select a patch maximum extension length of 0.5 as our experimental setting, corresponding to $\frac{D}{b} = \frac{D}{4}$.

Efficient Analysis

We conduct a comparison between HDMixer and well-established models by forecasting performance, training speed, and memory usage. Results are obtained using the official model configuration. The model efficiency is visualized in Fig.5 on the ETTh2.

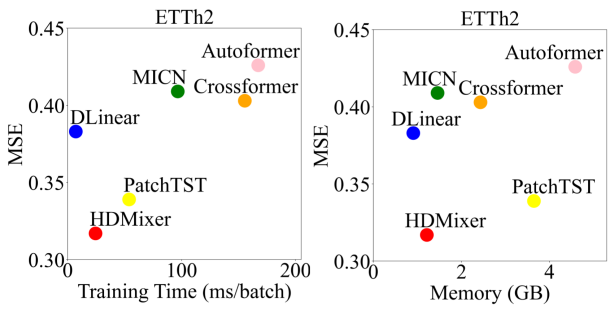

Figure 5: Model efficiency comparison of different methods on ETTh2 with 192 output length.

Acknowledgements

This paper is partially supported by the National Natural Science Foundation of China (No.62072427, No.12227901), the Project of Stable Support for Youth Team in Basic Research Field, CAS (No.YSBR-005), Academic Leaders Cultivation Program, USTC, and the grant from State Key Laboratory of Resources and Environmental Information System.

References

Chen, Y.; Chen, S.; Yang, Z.-X.; and Wu, E. 2024. Learning self-target knowledge for few-shot segmentation. *Pattern Recognition*, 110266.

Cheng, D.; Yang, F.; Xiang, S.; and Liu, J. 2022. Financial time series forecasting with multi-modality graph neural network. *Pattern Recognition*, 121: 108218.

Fang, Z.; Chen, Y.; Wang, Y.; Wang, Z.; Ji, X.; and Zhang, Y. 2023. Weakly-Supervised Semantic Segmentation for Histopathology Images Based on Dataset Synthesis and Feature Consistency Constraint. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(1): 606–613.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2015. Deep Residual Learning for Image Recognition. arXiv:1512.03385.

Hewamalage, H.; Bergmeir, C.; and Bandara, K. 2021. Recurrent neural networks for time series forecasting: Current status and future directions. *International Journal of Forecasting*, 37(1): 388–427.

Huang, Q.; Shen, L.; Zhang, R.; Ding, S.; Wang, B.; Zhou, Z.; and Wang, Y. 2023. CrossGNN: Confronting Noisy Multivariate Time Series Via Cross Interaction Refinement. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Kim, T.; Kim, J.; Tae, Y.; Park, C.; Choi, J.-H.; and Choo, J. 2021. Reversible instance normalization for accurate time-series forecasting against distribution shift. In *International Conference on Learning Representations*.

Lin, S.; Lin, W.; Wu, W.; Wang, S.; and Wang, Y. 2023a. PETformer: Long-term Time Series Forecasting via Placeholder-enhanced Transformer. arXiv:2308.04791.

Lin, S.; Lin, W.; Wu, W.; Zhao, F.; Mo, R.; and Zhang, H. 2023b. SegRNN: Segment Recurrent Neural Network for Long-Term Time Series Forecasting. arXiv:2308.11200.

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

Liu, Y.; Wu, H.; Wang, J.; and Long, M. 2022. Non-stationary Transformers: Exploring the Stationarity in Time Series Forecasting. In *Advances in Neural Information Processing Systems*.

Luo, D.; and Wang, X. 2023. Cross-LKTCN: Modern Convolution Utilizing Cross-Variable Dependency for Multivariate Time Series Forecasting Dependency for Multivariate Time Series Forecasting. arXiv:2306.02326.

Miao, H.; Zhao, Y.; Guo, C.; Yang, B.; Kai, Z.; Huang, F.; Xie, J.; and Jensen, C. S. 2024. A Unified Replay-based Continuous Learning Framework for Spatio-Temporal Prediction on Streaming Data. In *ICDE*.

Miao, H.; Zhong, X.; Liu, J.; Zhao, Y.; Zhao, X.; Qian, W.; Zheng, K.; and Jensen, C. S. 2023. Task Assignment with Efficient Federated Preference Learning in Spatial Crowdsourcing. *TKDE*.

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

Pincus, S. M. 1991. Approximate entropy as a measure of system complexity. *Proceedings of the National Academy of Sciences*, 88(6): 2297–2301.

Shabani, A.; Abdi, A.; Meng, L.; and Sylvain, T. 2023. Scaleformer: Iterative Multi-scale Refining Transformers for Time Series Forecasting. *The Eleventh International Conference on Learning Representations*.

Shang, C.; Chen, J.; and Bi, J. 2021. Discrete graph structure learning for forecasting multiple time series. *arXiv preprint arXiv:2101.06861*.

Shen, B.; Lin, Z.; Liu, Y.; Liu, Z.; Wang, L.; and Wang, W. 2022. COST-EFF: Collaborative Optimization of Spatial and Temporal Efficiency with Slenderized Multi-exit Language Models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, 1719–1730. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics.

Tang, T.; Lu, J.; and Qiu, J. 2023a. On Controllability and Stabilization of Switched Logical Control Networks with Switching-Input- Triggered Restricted Successors. *IEEE Transactions on Control of Network Systems*, 1–12.

Tang, T.; Lu, J.; and Qiu, J. 2023b. Robust Finite-Time Consensus of Coupled Markovian Logical Dynamic Networks. *IEEE Transactions on Circuits and Systems II: Express Briefs*.

Tian, H.; Xu, X.; Qi, L.; Zhang, X.; Dou, W.; Yu, S.; and Ni, Q. 2021. CoPace: Edge Computation Offloading and Caching for Self-Driving With Deep Reinforcement Learning. *IEEE Transactions on Vehicular Technology*, 70(12): 13281–13293.

Tolstikhin, I.; Houlsby, N.; Kolesnikov, A.; Beyer, L.; Zhai, X.; Unterthiner, T.; Yung, J.; Steiner, A.; Keysers, D.; Uszkoreit, J.; Lucic, M.; and Dosovitskiy, A. 2021. MLP-Mixer: An all-MLP Architecture for Vision. arXiv:2105.01601.

Wang, B.; Zhang, Y.; Shi, J.; Wang, P.; Wang, X.; Bai, L.; and Wang, Y. 2023a. Knowledge Expansion and Consolidation for Continual Traffic Prediction With Expanding Graphs. *IEEE Transactions on Intelligent Transportation Systems*.

Wang, B.; Zhang, Y.; Wang, P.; Wang, X.; Bai, L.; and Wang, Y. 2023b. A Knowledge-Driven Memory System for Traffic Flow Prediction. In *International Conference on Database Systems for Advanced Applications*, 192–207. Springer.

Wang, B.; Zhang, Y.; Wang, X.; Wang, P.; Zhou, Z.; Bai, L.; and Wang, Y. 2023c. Pattern Expansion and Consolidation on Evolving Graphs for Continual Traffic Prediction. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2223–2232.

Wang, H.; Peng, J.; Huang, F.; Wang, J.; Chen, J.; and Xiao, Y. 2023d. MICN: Multi-scale Local and Global Context

Modeling for Long-term Series Forecasting. In *The Eleventh International Conference on Learning Representations*.

Wang, P.; Ge, C.; Zhou, Z.; Wang, X.; Li, Y.; and Wang, Y. 2021. Joint gated co-attention based multi-modal networks for subregion house price prediction. *IEEE Transactions on Knowledge and Data Engineering*.

Wang, X.; Chen, L.; Zhang, H.; Wang, P.; Zhou, Z.; and Wang, Y. 2023e. A Multi-graph Fusion Based Spatiotemporal Dynamic Learning Framework. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*, 294–302.

Wang, X.; Zhang, H.; Wang, P.; Zhang, Y.; Wang, B.; Zhou, Z.; and Wang, Y. 2023f. An Observed Value Consistent Diffusion Model for Imputing Missing Values in Multivariate Time Series. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2409–2418.

Woo, G.; Liu, C.; Sahoo, D.; Kumar, A.; and Hoi, S. 2022. Etsformer: Exponential smoothing transformers for time-series forecasting. *arXiv preprint arXiv:2202.01381*.

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, Z.; Pan, S.; Long, G.; Jiang, J.; Chang, X.; and Zhang, C. 2020. Connecting the Dots: Multivariate Time Series Forecasting with Graph Neural Networks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*.

Wu, Z.; Pan, S.; Long, G.; Jiang, J.; and Zhang, C. 2019. Graph wavenet for deep spatial-temporal graph modeling. *arXiv preprint arXiv:1906.00121*.

Xu, X.; Huang, Q.; Zhu, H.; Sharma, S.; Zhang, X.; Qi, L.; and Bhuiyan, M. Z. A. 2021. Secure Service Offloading for Internet of Vehicles in SDN-Enabled Mobile Edge Computing. *IEEE Transactions on Intelligent Transportation Systems*, 22(6): 3720–3729.

Xu, X.; Zhang, X.; Gao, H.; Xue, Y.; Qi, L.; and Dou, W. 2020. BeCome: Blockchain-Enabled Computation Offloading for IoT in Mobile Edge Computing. *IEEE Transactions on Industrial Informatics*, 16(6): 4187–4195.

Yang, K.; Zhou, Z.; Sun, W.; Wang, P.; Wang, X.; and Wang, Y. 2023a. EXTRACT and REFINE: Finding a Support Subgraph Set for Graph Representation. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2953–2964.

Yang, Y.; Zhang, C.; Zhou, T.; Wen, Q.; and Sun, L. 2023b. DCdetector: Dual Attention Contrastive Representation Learning for Time Series Anomaly Detection. In *Proc. 29th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD 2023)*.

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

Yu, B.; Yin, H.; and Zhu, Z. 2018. Spatio-temporal Graph Convolutional Networks: A Deep Learning Framework for Traffic Forecasting. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI)*.

Yu, C.; Wang, F.; Shao, Z.; Sun, T.; Wu, L.; and Xu, Y. 2023. DSformer: A Double Sampling Transformer for Multivariate Time Series Long-term Prediction. arXiv:2308.03274.

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

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

Zhang, Y.; Lu, W.; Wang, X.; Wang, P.; and Wang, Y. 2023. Pondering About Task Spatial Misalignment: Classification-Localization Equilibrated Object Detection. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 1–5. IEEE.

Zhang, Y.; Wang, B.; Shan, Z.; Zhou, Z.; and Wang, Y. 2022b. CMT-Net: A mutual transition aware framework for taxicab pick-ups and drop-offs co-prediction. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, 1406–1414.

Zhang, Y.; and Yan, J. 2023. Crossformer: Transformer Utilizing Cross-Dimension Dependency for Multivariate Time Series Forecasting. In *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 *Proc. 39th International Conference on Machine Learning (ICML 2022)*.

Zhou, Z.; Huang, Q.; Lin, G.; Yang, K.; BAI, L.; and Wang, Y. 2023a. GReTo: Remedying dynamic graph topology-task discordance via target homophily. In *The Eleventh International Conference on Learning Representations*.

Zhou, Z.; Huang, Q.; Yang, K.; Wang, K.; Wang, X.; Zhang, Y.; Liang, Y.; and Wang, Y. 2023b. Maintaining the Status Quo: Capturing Invariant Relations for OOD Spatiotemporal Learning. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 3603–3614.

Zhou, Z.; Wang, Y.; Xie, X.; Chen, L.; and Liu, H. 2020a. RiskOracle: A minute-level citywide traffic accident forecasting framework. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, 1258–1265.

Zhou, Z.; Wang, Y.; Xie, X.; Chen, L.; and Zhu, C. 2020b. Foresee urban sparse traffic accidents: A spatiotemporal multi-granularity perspective. *IEEE Transactions on Knowledge and Data Engineering*, 34(8): 3786–3799.