2025 International Conference on Intelligent Computing
July 26-29, Ningbo, China
https://www.ic-icc.cn/2025/index.php

DAMLP: Data Augmented Multi-Layer Perceptrons for Multivariate Time Series Forecasting

Jiyanglin Li[1,2], Heming Du[3], Yiming Tang[4], Jinhong You[5], Shouguo Du[6], and Wen Li[7*]

[1] Guizhou Key Laboratory of Big Data Statistical Analysis, Guizhou University of Finance and Economics, Guizhou, China
[2] School of Mathematics and Statistics, Guizhou University of Finance and Economics, Guizhou, China
[3] School of Electrical Engineering and Computer Science, The University of Queensland, Queensland, Australia
[4] School of Statistics and Mathematics, Shanghai Lixin University of Accounting and Finance, Shanghai, China
[5] School of Statistics and Data Science, Shanghai University of Finance and Economics, Shanghai, China
[6] Shanghai Municipal Big Data Center, Shanghai, China
[7] School of Statistics and Information, Shanghai University of International Business and Economics, Shanghai, China
wen.li.sh@outlook.com

Abstract. Multivariate time series forecasting (MTSF) plays a crucial role in various applications by predicting future values based on historical data across multiple variates. Although deep learning models have achieved remarkable success in MTSF tasks, they often face challenges related to data scarcity. Data augmentation, which enriches the training data, has emerged as a promising technique to improve forecasting accuracy. However, preserving temporal dependencies in augmented data remains a significant challenge. In this paper, we introduce Data Augmented Multi-Layer Perceptrons (DAMLP), a novel MTSF framework that integrates a Data Augmentation (DA) module and a simple yet effective Multi-Layer Perceptrons (MLP) architecture. Our DA module enhances the training dataset by increasing the frequency of time series with high correlations to others while reducing the frequency of low-correlation series, thus mitigating the interference on the model's forecasting accuracy caused by low-correlation series. To efficiently utilize the augmented dataset, we use a simple MLP architecture that provides an efficient solution without sacrificing forecasting performance. Our experimental results on multiple real-world datasets demonstrate that DAMLP outperforms state-of-the-art models with less memory usage and training time. Our approach highlights the potential of leveraging correlation information to improve the accuracy and efficiency of MTSF models.

Keywords: Multivariate Time Series Forecasting, Data Augmentation, MLP.

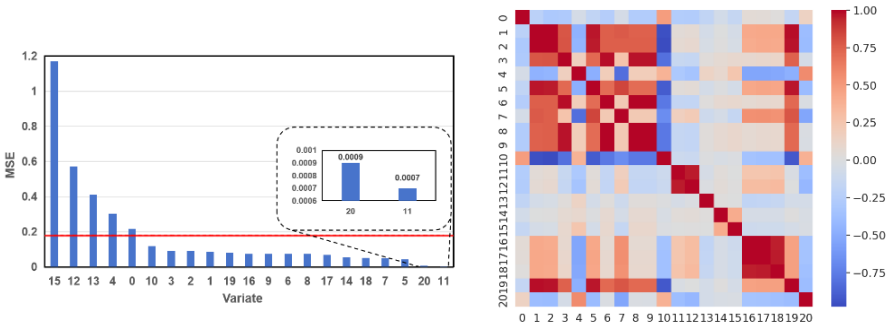
*Corresponding Author.

1 Introduction

Multivariate time series forecasting (MTSF) is a task that predicts future values based on sequential historical data of multiple variates, which has a wide range of applications in various fields such as [1-9]. In the past, traditional statistical models like ARIMA [10] and ARCH [11] were commonly utilized for time series forecasting. These models are based on specific assumptions and are simple, intuitive, and useful for identifying basic patterns in data. However, multivariate time series exhibit complex inter-series relationships and temporal dependencies, which makes MTSF task highly challenging and exceeding the capabilities of classical statistical methods. Recently, the success of Deep Learning has led to many deep time series forecasting models that can flexibly learn the complex relationships within time series and achieve remarkable results in MTSF tasks. These include RNN-based models [12, 13], CNN-based models [14-16], Transformer-based models [17-20], MLP-based models [21-23], Linear-based models [24-26], etc.

All of these deep forecasting models rely heavily on data and their performance often improves as the size of the training dataset increases [9, 27-29]. However, compared to fields like Computer Vision (CV) and Natural Language Processing (NLP), time series datasets are constrained by their limited available data with shorter timesteps and fewer variables. Therefore, as a critical strategy for enriching time series datasets, data augmentation is a promising solution to enhance the forecasting accuracy [5, 7, 30]. One of the primary challenges in time series data augmentation is how to preserve the temporal dependencies. Data augmentation techniques must ensure that synthetic data maintains these dependencies accurately [31]. If the temporal structures are not adequately preserved, the performance of forecasting models may degrade [32]. On the other hand, the synthetic data should closely resemble real-world observations to ensure that models trained on augmented data generalize effectively. Poorly generated synthetic data can introduce biases and inaccuracies, leading to unreliable model outcomes [33].

Therefore, we directly utilize the original series to augment the dataset without modifying the original data through methods such as jittering or scaling, ensuring the quality of the augmented data while preserving temporal dependencies. Specifically, to directly utilize the original data for data augmentation, it is crucial to appropriately leverage the relationships between different series. Many past methods trained models directly on the original datasets where all series appear with the same frequency, in such cases the low-correlation series interfere with the forecasting performance of the high-correlation series, thereby degrading the overall performance of the model. For example, Fig. 1a shows the prediction results of iTransformer [20] for each series in the weather dataset. The MSEs of the five series with low correlations to others (series 15, 12, 13, 4, and 0) are much higher than the average MSE, while the MSEs of series that are highly correlated with other series are lower than the average MSE. This result implies that low-correlation series disrupt the forecasting performance of high-correlation series. Inspired by this example, we should adjust the frequencies of each series in the augmented dataset to reduce the influence of the low-correlation series, which is helpful to improve the overall forecasting performance.



(a) MSE of each variate in the Weather dataset.

(b) The correlation of series in the Weather dataset.

Fig. 1. The prediction performance of iTransformer is interfered by the low-correlated series in the Weather dataset. (a) Displays the MSE of each variate on the test set (blue bars) along with the mean MSE (red horizontal line), with the results for series 20 and 11 zoomed in for clarity. These two series exhibit much lower MSEs compared to the others, as outliers cause the values of these series to approach zero after normalization, despite their weak correlations with the remaining series. (b) Shows the Pearson correlation coefficient of each pair of series.

On the other hand, the model architecture often involves a trade-off between accuracy and performance. For instance, Transformer-based models can learn more complex temporal dependencies through self-attention mechanisms, but they require more memory usage and training time. This limitation hinders their ability to utilize longer historical information for future predictions. Conversely, linear models [24-26] fail to capture intricate nonlinear relationships although they have fewer parameters and are more efficient. Consequently, their performances in MTSF with complex relationships are inferior to those of more sophisticated models.

Based on the above motivations, in this paper, we propose a forecasting framework to balance the forecasting accuracy and computational efficiency. Technically, we propose Data Augmented Multi-Layer Perceptrons (DAMLP) with a Data Augmentation (DA) module that enhances the training data without additional transformations of the original data. This module increases the frequency of time series with higher correlations to other variates during training while reducing the frequency of those with lower correlations, thereby mitigating the interference on the model's forecasting accuracy caused by low-correlation series. To efficiently handle the augmented dataset, we employ a MLP architecture as the core module for the series representation. The experimental results confirm the proposed DAMLP achieves superior performance on real-world forecasting datasets. Our main contributions are as follows:

1. We propose a simple but effective Data Augmentation (DA) module that does not modify the original data, ensuring the quality of the augmented data while preserving temporal dependencies. By adjusting the frequencies of each series in the training

process, this module helps mitigate the interference on the model's forecasting accuracy caused by low-correlation series.
2. We propose a MTSF framework called DAMLP. It takes a simple MLP architecture as its core module for the series representation, which is quite simple but can handle the augmented training data more efficiently while preserving the high forecasting accuracy.
3. We conduct extensive experiments on eight multivariate datasets and the results show that DAMLP outperforms state-of-the-art baselines. Our method achieves impressive results with less memory usage and training time than many mainstream models.

2 Related Work

2.1 Time Series Forecasting

Deep learning methods have gained prominence in time series forecasting due to their ability to capture complex time dependencies. These include RNN-based models like DeepAR [12] and WITRAN [13], CNN-based models like SCINet [14], TimesNet [15], and ModernTCN [16], Transformer-based models like Autoformer [17], FEDformer [18], PatchTST [19], iTransformer [20], etc. However, these methods have several limitations. CNN-based models struggle to capture long-term dependencies due to their limited receptive field. RNN-based models are inefficient at utilizing training resources because of their inability to parallelize computations. Although Transformer-based models exhibit powerful forecasting performance, their quadratic computational and memory complexity results in significant memory overhead and slow inference speeds.

Recently, many studies have adopted linear-based models with promising results including DLinear [24], RLinear [25], SparseTSF [26], etc. However, Linear-based models face limitations when applied to multivariate time series with complex temporal dependencies. On the other hand, MLP-based models can implicitly learn nonlinear relationships, and they often achieve comparable performance to more complex models, while significantly reducing computational costs. For example, Timemixer [21] combines multi-scale information to handle intricate temporal variations, U-Mixer [23] adopts the U-Net architecture to merge low- and high-level features, resulting in more comprehensive data representations, and FITS [22] manipulates the series through interpolation in the complex frequency domain. These studies highlight the effectiveness of MLP-based networks in time series forecasting tasks and inspire us to take MLP as our main module to learn temporal dependencies more efficiently.

2.2 Data Augmentation

Data augmentation, as an effective way to enhance the size and quality of the training data, is an important technique to improve the performance of deep time series forecasting models [30]. The transformations of the original series are the most straightforward data augmentation methods for time series data. The window slicing method [34] is simple and effective for extracting segments of the time series, but it may not capture

complex temporal dependencies. Window warping [35] generates new samples by compressing or extending specific time ranges, but it requires careful handling to maintain data consistency. Noise injection [36] improves model robustness by adding small amounts of noise, but it must be carefully adjusted to avoid introducing artifacts.

In this paper, different from the methods mentioned above, we propose a simple yet effective data augmentation method for the training data without modifying the original series such that our training process can focus more on series with stronger correlations with others.

3 Methods

3.1 Problem Definition

In time series forecasting tasks, given a collection of multivariate time series samples with look-back window L: $\boldsymbol{X} = (\boldsymbol{x}_{1,:}, \ldots, \boldsymbol{x}_{L,:})^\top = (\boldsymbol{x}_{:,1}, \ldots, \boldsymbol{x}_{:,D}) \in \mathbb{R}^{L \times D}$ where each $\boldsymbol{x}_{t,:} \in \mathbb{R}^D$ at time step t is a vector of dimension D and $\boldsymbol{X}_{:,i} \in \mathbb{R}^L$ is the i-th variate of the D historical series, the goal of the forecasting model is to predict the H future values $\boldsymbol{Y} = (\boldsymbol{x}_{L+1,:}, \ldots, \boldsymbol{x}_{L+H,:})^\top \in \mathbb{R}^{H \times D}$ where each $\boldsymbol{x}_{L+t,:}$ at time step $L + t$ is a vector of dimension D and $\top$ is the transpose symbol for matrix and vector. We also define $\boldsymbol{X}_{train} = (X_1, \cdots, X_D) \in \mathbb{R}^{T \times D}$ as the training data, where T is the length of the series in the training data, and for $d = 1, \cdots, D$, $X_d \in \mathbb{R}^T$ is the d-th series with length T.

3.2 Correlation Coefficients

In statistics, Pearson Correlation Coefficient (PCC) [37] is one of the correlation coefficients that measures correlation between two sets of data. When applied to a population (meaning the case in which we treat variates X and Y as random variables), it is defined as follows.

Definition 1. Given a pair of random variables (X, Y), the Pearson correlation coefficient is defined as:

$$\rho(X, Y) = \frac{Cov(X,Y)}{\sqrt{Var(X)Var(Y)}}, \tag{1}$$

where $Cov(X, Y) = E[(X - E(X))(Y - E(Y))]$ is the covariance of X and Y, $Var(X) = E(X - E(X))^2$ is the variance of X, and $Var(Y) = E(Y - E(Y))^2$ is the variance of Y.

When applied to the case of time series sample, it is commonly represented by r_{xy} and may be referred to as the sample correlation coefficient or the sample Pearson correlation coefficient, which is defined as follows.

Definition 2. Given a pair of time series data $x = \{x_t\}_{t=1}^T$ and $y = \{y_t\}_{t=1}^T$ of length T, the sample Pearson correlation coefficient is defined as:

$$r_{xy} = \frac{\sum_{t=1}^T (x_t - \bar{x})(y_t - \bar{y})}{\sqrt{\sum_{t=1}^T (x_t - \bar{x})^2 \sum_{t=1}^T (y_t - \bar{y})^2}}, \tag{2}$$

where $\bar{x} = \frac{1}{T}\sum_{t=1}^{T} x_t$ is the sample mean of $\{x_t\}_{t=1}^{T}$, and analogously for $\bar{y}$.

In a multivariate time series dataset where x_i and x_j represent the i-th and j-th series respectively, we also denote $r_{ij} = r_{x_i x_j}$ as the correlation coefficient of x_i and x_j.

3.3 DAMLP

Fig. 2 illustrates the overall architecture of our work for time series forecasting, comprising four core components: DA, a data augmentation module; Norm and DeNorm [37], the reversible normalization and de-normalization as a pair to alleviate the distribution shift problem; MLP, a core module to learn the time dependencies of the series; Projection, a linear layer that projects the output of MLP to the forecasting feature space.

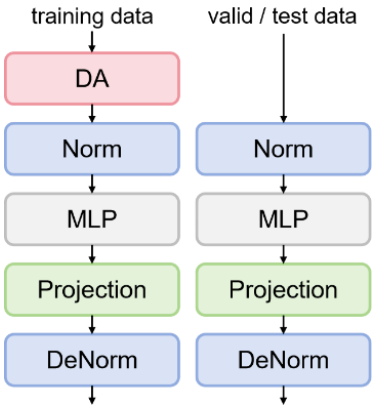


Fig. 2. The overall architecture of our work for time series forecasting. The pipeline comprises DA, Norm, MLP, Projection, and DeNorm for the training data while the DA module will be removed for the valid and test data.

Data Augmentation Module. As shown in Fig. 2, for the training data, our DAMLP pipeline starts with the Data Augmentation (DA) module, which refines the input data by focusing on the most relevant inter-series relationships. It is also shown in Fig. 2 that the validation and test data will not be transformed through the DA module since the former is only used to determine when to stop training to prevent over-fitting, while the latter is used to evaluate the model's performance. The details about the DA module are shown in Fig. 3 and the process of DA can be formulated as follows:

$$\boldsymbol{X}'_{train} = NeighborFinder(\boldsymbol{X}_{train}), \tag{3}$$

$$\boldsymbol{x}_{train}, \boldsymbol{y}_{train} = IndependentSlidingWindow(\boldsymbol{X}'_{train}). \tag{4}$$

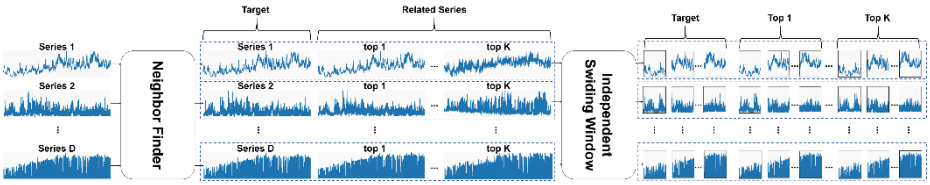


Fig. 3. Data Augmentation Module.

The *NeighborFinder* is used to find the most related series for each series and place these series behind the original series one by one, and the *IndependentSlidingWindow* is to split each series into sliding windows. To be more specific, by calculating the correlation coefficient $r_{ij}(j = 1, \cdots, D, j \neq i)$ for each series $X_i \in \boldsymbol{X}_{train}$ and ordering them by their absolute value in descending order, we can obtain the k most related series $\{X_i^{(j)}\}_{j=1}^{k}$ for each series, where $X_i^{(j)} \in \mathbb{R}^T$ denotes the series that is the j-th most correlated with X_i. Then we place these series one after another behind the original series to obtain a new series $X_{i'} = (X_i^\top, X_i^{(1)\top}, \cdots, X_i^{(k)\top})^\top \in \mathbb{R}^{T(k+1)}$. After processing each series in this way, we obtain the augmented training dataset $\boldsymbol{X}'_{train} = (X_{1'}, \cdots, X_{D'}) \in \mathcal{R}^{T(k+1)\times D}$, which is the output of the *NeighborFinder*. Then we take the *IndependentSlidingWindow* to split each series in $X_{i'}$ (i.e., $X_i^\top$ and $X_j^{(1)\top}, j = 1, \cdots, k$) into S subseries, with each containing a look-back and a horizon window, which results in an input $\boldsymbol{\mathcal{X}}_{train} = \{\boldsymbol{X}^1, \cdots, \boldsymbol{X}^{S(k+1)}\} \in \mathbb{R}^{S(k+1)\times L\times D}$ and an output $\boldsymbol{\mathcal{Y}}_{train} = \{\boldsymbol{Y}^1, \cdots, \boldsymbol{Y}^{S(k+1)}\} \in \mathbb{R}^{S(k+1)\times H\times D}$, where $S = T - L - H + 1$, L and H are the look-back window length and horizon window length respectively, and $\boldsymbol{X}^i \in \mathbb{R}^{L\times D}$ and $\boldsymbol{Y}^i \in \mathbb{R}^{H\times D}$ are the input and output of the i-th sample respectively.

By this dedicatedly designed DA module, the series that are more correlated with others appear more frequently while those with lower relevance appear less often in the subsequent training phase, which is capable of mitigating the interference of the low-correlation series for the overall forecasting performance.

Norm and DeNorm. Before the series are fed into the MLP module, we take RevIN [38], a simple yet effective and model-agnostic technique, to normalize the input in the look-back window. The reversed transformation is also used to denormalize the output of the projection layer to get the final prediction result. The normalization and de-normalization mitigate the distribution shift effect between the training and testing data, corresponding to the Norm and DeNorm module in Fig. 2 respectively, which are formulated as follows:

$$\widetilde{\boldsymbol{X}}^i = RevIN(\boldsymbol{X}^i), \tag{5}$$

$$\widehat{\boldsymbol{Y}}^i = RevIN^{-1}(\widetilde{\boldsymbol{Y}}^i), \tag{6}$$

where $\widetilde{\boldsymbol{X}}^i \in \mathbb{R}^{L\times D}$ is the normalized input in the i-th look-back window, $\widehat{\boldsymbol{Y}}^i \in \mathbb{R}^{L\times D}$ is the denormalized final forecasting result, $\widetilde{\boldsymbol{Y}}^i \in \mathbb{R}^{L\times D}$ is the output of the projection

layer introduced in the following subsection, $RevIN(\cdot)$ denotes the instance normalization operation, and $RevIN^{-1}(\cdot)$ is the inverse transformation of $RevIN(\cdot)$ with the same parameters.

MLP and Projection layer. We separate the MLP and the projection layer into two parts, so we can easily stack multiple MLPs to extend the network or replace the MLP with other blocks such as CNN or Transformer without modifying the projection layer if necessary, which makes our framework more flexible and user-friendly.

The MLP layer serves as the core module for feature extraction, processing the augmented input data to learn effective series representations. By introducing nonlinear activation functions, the MLP layer is capable of capturing complex temporal dependencies within multivariate time series, thus maintaining sufficient capacity to identify meaningful patterns in the data while achieving highly-efficient training and inference. Additionally, the MLP architecture can easily accommodate different task requirements by adjusting the number of layers or the hidden dimension size. The MLP module consists of two linear layers with an intermediate RELU activation function, which is formulated as follows:

$$\boldsymbol{Z}^i = MLP(\widetilde{\boldsymbol{X}}^i) = \boldsymbol{W}_2\sigma(\boldsymbol{W}_1\widetilde{\boldsymbol{X}}^i + \boldsymbol{b_1}) + \boldsymbol{b}_2, \tag{7}$$

where $\boldsymbol{Z}^i \in \mathbb{R}^{L\times D}$ denotes the output of the MLP, $\boldsymbol{W}_1 \in \mathbb{R}^{d\times L}$, $\boldsymbol{W}_2 \in \mathbb{R}^{L\times d}$, $\boldsymbol{b}_1 \in \mathbb{R}^d$, $\boldsymbol{b}_2 \in \mathbb{R}^L$, d is the dimension of the hidden layer, and $\sigma(\cdot)$ is the point-wise RELU activation function. Note that the bias terms $\boldsymbol{b}_1$ and $\boldsymbol{b}_2$ are 1-dimensional vectors, but they will be broadcasted into $\mathbb{R}^{L\times D}$ along the second dimension during the addition operation to enable the correct execution of the subsequent calculations.

The Projection layer maps the output of the MLP to the forecasting feature space. This ensures that the predictions are aligned with the length of the horizon window. It is a linear layer formulated as follows:

$$\widetilde{\boldsymbol{Y}}^i = Linear(\boldsymbol{Z}^i) = \boldsymbol{W}_3\boldsymbol{Z}^i + \boldsymbol{b}_3, \tag{8}$$

where $\boldsymbol{W}_3 \in \mathbb{R}^{H\times L}$, and $\boldsymbol{b}_3 \in \mathbb{R}^H$ is a 1-dimensional vector for the same reason as $\boldsymbol{b}_1$ and $\boldsymbol{b}_2$.

4 Experiments

4.1 Experimental Settings

Datasets. We evaluate the forecasting performance of our model on eight popular real-world datasets, including Electricity (ECL), Traffic, Weather, Solar, and 4 PEMS datasets (PEMS03, PEMS04, PEMS07, PEMS08) used by iTransformer [20].
Table **1** provides statistical information of those eight real-world datasets.

Table 1. Detailed dataset descriptions.

Dataset	# of variates	Timesteps	Data Partition	Frequency
ECL	321	26304	7:1:2	Hourly
Traffic	862	17544	7:1:2	Hourly
Weather	21	52696	7:1:2	10min
Solar	137	52560	7:1:2	10 min
PEMS03	358	26208	6:2:2	5 min
PEMS04	307	16992	6:2:2	5 min
PEMS07	883	28224	6:2:2	5 min
PEMS08	170	17856	6:2:2	5 min

Evaluation metrics. Following previous works [20], we use Mean Squared Error (MSE) and Mean Absolute Error (MAE), which are separately computed between the prediction and the ground truth on the test set as the evaluation metric for our multivariate time series forecasting task.

Baselines. We select SOTA Transformer-based models for MTSF including iTransformer [20] and Crossformer [39]. We also incorporate TiDE [40], which is one of the SOTA MLP-based models, for comparison purposes. Recently, DLinear [24] demonstrated that simple linear models can outperform these complex methods, hence we take DLinear as an important baseline. Besides, we also conduct a comparison with TimesNet [15], a CNN-based model, to further evaluate the performance of our method compared to different types of models.

Experimental Setup. All experiments in this study were implemented using PyTorch [41] and conducted using an NVIDIA GeForce RTX 3090 24GB GPU. We utilize ADAM [42] with L2 loss for the model optimization. We set the same historical look-back window size of $L = 336$ for all models on all datasets. The predicting length is set to $H \in \{96,192,336,720\}$. The hyperparameters of our model are obtained from the grid search. To be specific, we explore the learning rate within the set $\{0.01,0.001,0.0005,0.0001\}$, the dropout rate within $\{0.1,0.2,0.3,0.5\}$, the batch size within $\{16,32,64,128\}$, and the hidden dimension d within $\{32,64,128,256,512\}$, and the layers within $\{1,2,3,4\}$. Considering the dataset size and computational resources, we set the training epochs to 10 for the short-term forecasting (PEMS) datasets and to 30 for the other datasets.

Additionally, we apply an early-stopping strategy with a patience of 5 epochs to prevent overfitting, ensuring that the final model generalizes well to unseen data.

4.2 Main Results

Table 2 shows the multivariate forecasting results. Overall, our model outperforms all baseline methods in almost all the cases. Quantitatively, compared with the DLinear model, DAMLP achieves an overall 44.6% reduction in MSE and 34.4% on MAE ,

with even more significant improvement in short-term forecasting, as its MLP module can capture more complex nonlinear temporal dependencies. It also outperforms the TimesNet model with a 24.1% reduction in MSE and 14.6% in MAE. In addition, compared with Transformer-based models, DAMLP can still outperform them in general, especially for the longer horizon windows (ECL, Traffic, Weather, Solar). This phenomenon can be attributed to DAMLP's specialized Data Augmentation (DA) module, which strategically increases the representation frequency of highly correlated series, enabling the model to better capture stable long-term patterns and effectively mitigate interference from less relevant series. Consequently, DAMLP achieves robust and reliable long-horizon forecasting, reflecting its capability to leverage correlation information for enhanced predictive accuracy.

Table 2. Multivariate forecasting results with horizon $H \in \{12,24,48,96\}$ for PEMS and $H \in \{96,192,336,720\}$ for others. The best results are in bold and the second best are underlined.

Models		DAMLP (Ours)		iTransformer 2024		Crossformer 2023		TiDE 2023		TimesNet 2023		DLinear 2023	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ECL	96	**0.133**	**0.230**	0.148	0.240	0.219	0.314	0.237	0.329	0.168	0.272	0.197	0.282
	192	**0.152**	**0.247**	0.162	0.253	0.231	0.322	0.236	0.330	0.184	0.289	0.196	0.285
	336	**0.169**	**0.264**	0.178	0.269	0.246	0.337	0.249	0.344	0.198	0.300	0.209	0.301
	720	**0.208**	**0.298**	0.225	0.317	0.280	0.363	0.284	0.373	0.220	0.320	0.245	0.333
	Avg	**0.166**	**0.260**	0.178	0.270	0.244	0.334	0.251	0.344	0.192	0.295	0.212	0.300
Traffic	96	**0.375**	**0.262**	0.395	0.268	0.522	0.290	0.805	0.493	0.593	0.321	0.650	0.396
	192	**0.394**	**0.273**	0.417	0.276	0.530	0.293	0.756	0.474	0.617	0.336	0.598	0.370
	336	**0.405**	**0.277**	0.433	0.283	0.558	0.305	0.762	0.477	0.629	0.336	0.605	0.373
	720	**0.438**	**0.297**	0.467	0.302	0.589	0.328	0.719	0.449	0.640	0.350	0.645	0.394
	Avg	**0.403**	**0.277**	0.428	0.282	0.550	0.304	0.760	0.473	0.620	0.336	0.625	0.383
Weather	96	0.160	**0.208**	0.174	0.214	**0.158**	0.230	0.202	0.261	0.172	0.220	0.196	0.255
	192	**0.205**	**0.249**	0.221	0.254	0.206	0.277	0.242	0.298	0.219	0.261	0.237	0.296
	336	**0.257**	**0.290**	0.278	0.296	0.272	0.335	0.287	0.335	0.280	0.306	0.283	0.335
	720	**0.332**	**0.343**	0.358	0.347	0.398	0.418	0.351	0.386	0.365	0.359	0.345	0.381
	Avg	**0.239**	**0.272**	0.258	0.278	0.259	0.315	0.271	0.320	0.259	0.287	0.265	0.317
Solar	96	**0.194**	**0.244**	0.203	0.237	0.310	0.331	0.312	0.399	0.250	0.292	0.290	0.378
	192	**0.214**	**0.264**	0.233	0.261	0.734	0.725	0.339	0.416	0.296	0.318	0.320	0.398
	336	**0.229**	**0.274**	0.248	0.273	0.750	0.735	0.368	0.430	0.319	0.330	0.353	0.415
	720	**0.219**	**0.272**	0.249	0.275	0.769	0.765	0.370	0.425	0.338	0.337	0.356	0.413
	Avg	**0.214**	**0.263**	0.233	0.262	0.641	0.639	0.347	0.417	0.301	0.319	0.330	0.401
PEMS03	12	**0.063**	**0.167**	0.071	0.174	0.090	0.203	0.178	0.305	0.085	0.192	0.122	0.243
	24	**0.081**	**0.186**	0.093	0.201	0.121	0.240	0.257	0.371	0.118	0.223	0.201	0.317
	48	**0.108**	**0.212**	0.125	0.236	0.202	0.317	0.379	0.463	0.155	0.260	0.333	0.425
	96	**0.132**	**0.233**	0.164	0.275	0.262	0.367	0.490	0.539	0.228	0.317	0.457	0.515
	Avg	**0.096**	**0.200**	0.113	0.221	0.169	0.281	0.326	0.419	0.147	0.248	0.278	0.375
PEMS04	12	0.080	0.187	**0.078**	**0.183**	0.098	0.218	0.219	0.340	0.087	0.195	0.148	0.272
	24	0.097	**0.204**	**0.095**	0.205	0.131	0.256	0.292	0.398	0.103	0.215	0.224	0.340
	48	0.123	**0.232**	**0.120**	0.233	0.205	0.326	0.409	0.478	0.136	0.250	0.355	0.437
	96	**0.146**	**0.248**	0.150	0.262	0.402	0.457	0.492	0.532	0.190	0.303	0.452	0.504
	Avg	**0.111**	**0.218**	**0.111**	0.221	0.209	0.314	0.353	0.437	0.129	0.241	0.295	0.388
PEMS07	12	**0.058**	**0.162**	0.067	0.165	0.094	0.200	0.173	0.304	0.082	0.181	0.115	0.242
	24	**0.073**	**0.181**	0.088	0.190	0.139	0.247	0.271	0.383	0.101	0.204	0.210	0.329
	48	**0.094**	**0.202**	0.110	0.215	0.311	0.369	0.446	0.495	0.134	0.238	0.398	0.458
	96	**0.115**	**0.221**	0.139	0.245	0.396	0.442	0.628	0.577	0.181	0.279	0.594	0.553
	Avg	**0.085**	**0.191**	0.101	0.204	0.235	0.315	0.380	0.440	0.124	0.225	0.329	0.395
PEMS08	12	**0.078**	**0.175**	0.079	0.182	0.165	0.214	0.227	0.343	0.112	0.212	0.154	0.276
	24	**0.104**	**0.195**	0.115	0.219	0.215	0.260	0.318	0.409	0.141	0.238	0.248	0.353
	48	**0.141**	**0.217**	0.186	0.235	0.315	0.355	0.497	0.510	0.198	0.283	0.440	0.470
	96	**0.199**	**0.238**	0.221	0.267	0.377	0.397	0.721	0.592	0.320	0.351	0.674	0.565
	Avg	**0.131**	**0.206**	0.150	0.226	0.268	0.307	0.441	0.464	0.193	0.271	0.379	0.416

A closer examination of the results in Table 2 reveals three important insights: 1) in high-dimensional datasets such as ECL (321 variates) and Traffic (862 variates), DAMLP consistently outperforms other models. This suggests that DAMLP is particularly effective in mitigating noise and capturing stable global trends under high-dimensional conditions. 2) The DA module is also capable of reinforcing meaningful seasonal patterns by emphasizing strongly correlated variables, thereby enhancing long-term temporal features consistency. This is particularly beneficial for datasets characterized by clear or pronounced periodic behaviors — a property exhibited to varying degrees by all the datasets use in this paper. 3) In short-term traffic prediction datasets (PEMS), while DAMLP still achieves the lowest average errors in most forecasting horizons, its performance on PEMS04 is slightly inferior to that of iTransformer. We speculate that this may be due to the local nature of fluctuations, which could interfere with the results of inter-series correlation used in the data augmentation process. Nevertheless, all these results underscore the generalizability of the proposed method across both short- and long-term forecasting tasks, and these findings validate DAMLP's ability to adaptively emphasize relevant series through its correlation aware data augmentation, which in turn improves the model's representation of long-term dependencies and boosts its predictive accuracy across a wide range of multivariate time series.

In addition, the standard deviation of DAMLP's results is very small. We report the results under three runs with different random seeds in Table 3, which shows that the performance of DAMLP is stable. In most cases, the standard deviation across 5 runs is significantly small (within 0.001), which strongly indicates the robustness of DAMLP.

Table 3. Robustness of DAMLP performance under different horizon. The results are obtained from five random seeds.

Dataset	Metric	96	192	336	720
ECL	MSE	0.133±0.000	0.152±0.001	0.169±0.000	0.208±0.000
	MAE	0.231±0.001	0.248±0.001	0.265±0.001	0.297±0.001
Traffic	MSE	0.375±0.002	0.394±0.001	0.405±0.001	0.439±0.004
	MAE	0.263±0.002	0.272±0.002	0.278±0.001	0.299±0.004
Weather	MSE	0.160±0.000	0.205±0.000	0.257±0.001	0.332±0.001
	MAE	0.208±0.000	0.249±0.000	0.290±0.000	0.343±0.000
Solar	MSE	0.194±0.008	0.211±0.007	0.227±0.004	0.217±0.002
	MAE	0.244±0.005	0.262±0.006	0.270±0.003	0.269±0.003
PEMS03	MSE	0.063±0.000	0.082±0.001	0.109±0.001	0.133±0.001
	MAE	0.167±0.000	0.187±0.002	0.213±0.002	0.233±0.001
PEMS04	MSE	0.080±0.000	0.097±0.001	0.123±0.001	0.145±0.001
	MAE	0.186±0.001	0.204±0.001	0.231±0.002	0.248±0.002
PEMS07	MSE	0.059±0.001	0.073±0.001	0.094±0.001	0.114±0.001
	MAE	0.161±0.003	0.179±0.002	0.200±0.002	0.220±0.002
PEMS08	MSE	0.079±0.002	0.109±0.009	0.144±0.003	0.202±0.004
	MAE	0.176±0.002	0.201±0.008	0.220±0.003	0.242±0.004

4.3 Ablation Studies

To validate the soundness of our design in DAMLP, we perform an ablation study and report the results across all datasets. The results are shown in Table 4 and Table 5. Across all datasets, DAMLP consistently outperforms the version without the Data Augmentation (DA) module in terms of both MSE (Mean Squared Error) and MAE (Mean Absolute Error). The improvement brought by the DA module is consistent across diverse datasets, highlighting the robustness and generalizability of the DA module in enhancing forecasting performance for different types of multivariate time series.

Besides, from Table 5, we find that our DA module's impact is evident in both short and long prediction windows, but its effect is more significant for longer horizon windows. This effect may be explained by the fact that short-term predictions mainly concern local fluctuations, where data augmentation has less influence. In contrast, long-term predictions rely on more stable patterns, such as trends and seasonality. Our DA module provides the model with more reliable data to learn these long-term dependencies, thereby improving its performance for longer horizons.

Table 4. Ablation studies on ECL, Traffic, Solar, and Weather datasets.

Design	Horizon	ECL		Traffic		Solar		Weather	
		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
original	96	0.133	0.230	0.375	0.262	0.194	0.244	0.160	0.208
	192	0.152	0.247	0.394	0.273	0.214	0.264	0.205	0.249
	336	0.169	0.264	0.405	0.277	0.229	0.274	0.257	0.290
	720	0.208	0.298	0.438	0.297	0.219	0.272	0.332	0.343
	Avg	**0.166**	**0.260**	**0.403**	**0.277**	**0.214**	**0.263**	**0.239**	**0.272**
w/o DA	96	0.136	0.233	0.379	0.266	0.204	0.252	0.178	0.229
	192	0.152	0.247	0.401	0.278	0.218	0.265	0.220	0.264
	336	0.168	0.263	0.417	0.288	0.222	0.272	0.267	0.298
	720	0.209	0.298	0.449	0.308	0.234	0.282	0.335	0.345
	Avg	0.166	0.260	0.412	0.285	0.219	0.268	0.250	0.284

Table 5. Ablation studies on four PEMS datasets.

Design	Horizon	PEMS03		PEMS04		PEMS07		PEMS08	
		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
original	12	0.063	0.167	0.080	0.187	0.058	0.162	0.078	0.175
	24	0.081	0.186	0.097	0.204	0.073	0.181	0.104	0.195
	48	0.108	0.212	0.123	0.232	0.094	0.202	0.141	0.217
	96	0.132	0.233	0.146	0.248	0.115	0.221	0.199	0.238
	Avg	**0.096**	**0.200**	**0.111**	**0.218**	**0.085**	**0.191**	**0.131**	**0.206**
w/o DA	12	0.068	0.174	0.083	0.190	0.062	0.166	0.082	0.182
	24	0.086	0.192	0.104	0.213	0.083	0.192	0.112	0.208
	48	0.119	0.224	0.140	0.246	0.111	0.226	0.165	0.247
	96	0.146	0.244	0.169	0.272	0.135	0.248	0.229	0.280
	Avg	0.105	0.209	0.124	0.230	0.098	0.208	0.147	0.229

4.4 Model Analysis

The number of the most correlated series. The number of the most correlated series k plays a significant role in the Data Augmentation module. In this part, we conduct experiments on the Weather dataset to investigate the impact of k on forecasting performance, with the results shown in Fig. 4. The figure demonstrates that as k increases, there is a sharp metric downtrend at the very beginning followed by a gradual stabilization. However, the result for $k = 20$ is worse than that for $k = 19$. This is because when $k = 20$, the frequency of occurrence of each variate in the data-augmented training phase is exactly the same, as illustrated in Fig. 5. As a result, series with different correlation levels are learned the same number of times during training, which leads to poorer prediction results. This justifies our motivation to increase the frequencies of high-correlation series during training.

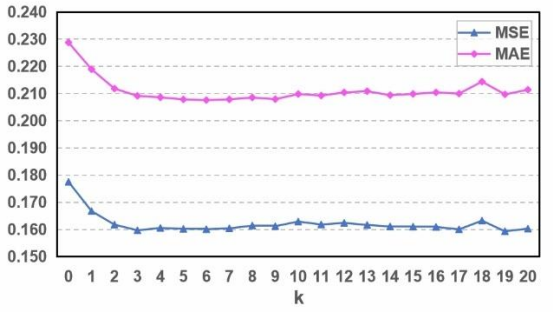


Fig. 4. MSE and MAE of filters under different bandwidths on the Weather dataset.

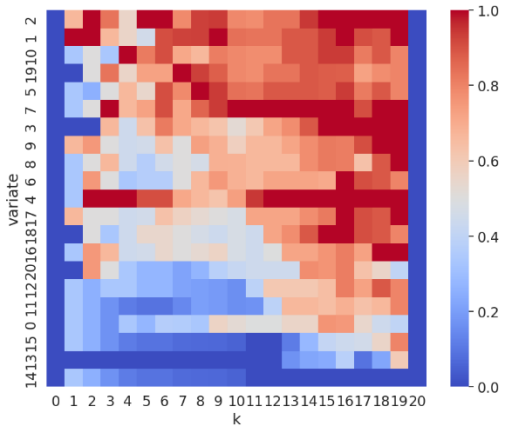


Fig. 5. The frequency with which each variate appears during training for each k. Each row corresponds to a variate, arranged in descending order of its correlation with other series. The k-th column represents the frequency at which each series appears during training when selecting the top-k series for data augmentation, with these frequencies normalized to the range [0, 1]. When $k = 0$ or $k = 20$, each series appears with the same frequency: once for $k = 0$ and 21 times for $k = 20$.

Therefore, our DA module offers a significant advantage over training models when $k > 0$. However, as k increases, the marginal effect of this advantage gradually diminishes. It is also important to note that the training cost with large k cannot be ignored in practical applications even when MLP- or Linear-based models are used, especially in high-dimensional datasets such as ECL and Traffic. Thus, according to our analysis above, it's reasonable and effective to choose $k = 5$ for these datasets and it can yield better forecasting results.

Efficiency Analysis. To comprehensively assess efficiency, we evaluate it along two dimensions: memory usage and training time. Specifically, we compare the efficiency of our DAMLP with the baselines using the ECL dataset, which contains 321 variables and 26,304 timestamps. The results, as shown in Fig. 6, highlight the advantages of DAMLP over the other models. While our approach demonstrates similar forecasting accuracy to iTransformer, its efficiency significantly outperforms iTransformer, with lower memory costs. In contrast, DLinear, although comparable to DAMLP in terms of memory usage and training time, exhibits much worse performance in forecasting accuracy. Therefore, DAMLP offers a similar speed and memory footprint as linear models while achieving superior forecasting performance.

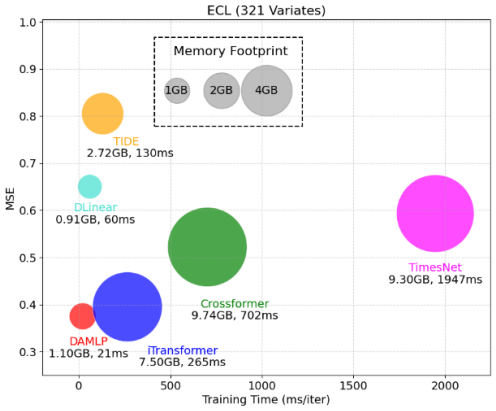


Fig. 6. Model effectiveness and efficiency comparison on the ECL datasets.

5 Conclusion

In this paper, we propose DAMLP, a novel multivariate time series forecasting framework that integrates a Data Augmentation (DA) module and a simple but efficient MLP architecture. By adjusting the frequencies of each series in training process with DA module, this module helps mitigate the interference on the model's forecasting accuracy caused by low-correlation series. DAMLP takes a simple MLP architecture, which

provides an efficient solution without sacrificing forecasting performance, to efficiently utilize the augmented dataset. Extensive experiments on real-world datasets demonstrate that DAMLP outperforms state-of-the-art models in both accuracy and efficiency, requiring less memory and training time. Our results highlight the importance of leveraging inter-series correlations in multivariate time series forecasting.

Acknowledgments. This research was supported by Shanghai Action Plan for Science, Technology and Innovation (No. 24BC3200404), Guizhou University of Finance and Economics Introduced Talents for Scientific Research (2022YJ029), Degree Program Development Initiative of Shanghai University of International Business and Economics (Research and Practice of Deep Learning in the Course of Time Series Analysis).

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

References

1. S. Lan, Y. Ma, W. Huang, W. Wang, H. Yang, and P. Li, “Dstagnn: Dynamic spatial-temporal aware graph neural network for traffic flow forecasting,” in International Conference on Machine Learning. PMLR, 2022, pp. 11 906–11 917.
2. T. Muhammad, A. B. Aftab, M. Ibrahim, M. M. Ahsan, M. M. Muhu, S. I. Khan, and M. S. Alam, “Transformer-based deep learning model for stock price prediction: A case study on bangladesh stock market,” International Journal of Computational Intelligence and Applications, vol. 22, no. 03, p. 2350013, 2023.
3. D. A. Gomez-Cravioto, R. E. Diaz-Ramos, F. J. Cantu-Ortiz, and H. G. Ceballos, “Data analysis and forecasting of the covid-19 spread: A comparison of recurrent neural networks and time series models,” Cognitive Computation, vol. 16, no. 4, pp. 1794–1805, 2024.
4. J. Wang, Y. Qian, L. Zhang, K. Wang, and H. Zhang, “A novel wind power fore- casting system integrating time series refining, nonlinear multi-objective optimized deep learning and linear error correction,” Energy Conversion and Management, vol. 299, p. 117818, 2024.
5. W. Li, W. Yu, H. Du, S. Du, J. You, and Y. Tang, “Learning seasonal-trend representations and conditional heteroskedasticity for time series analysis,” in International Conference on Artificial Neural Networks. Springer, 2024, pp. 267– 281.
6. W. Li, Y. Gu, and S. Du, “Tf-cl: Time series forcasting based on time-frequency domain contrastive learning,” in International Conference on Artificial Neural Networks. Springer, 2024, pp. 312–327.
7. L. Wen, C. Jiawei, L. Ruixue, H. Yuguo, and D. Shouguo, “T-transformer model for predicting tensor time series.” Journal of Computer Engineering & Applications, vol. 59, no. 11, 2023.
8. A. F. Alsayyad, A. A. M. A. Ali, M. Mabrouk, A. Al-Shammari, and M. Zrigui, "A survey on time series data classification: Blockchain technologies and security concerns," Procedia Computer Science, vol. 246, pp. 961–970, 2024.
9. H. Du, L. Li, Z. Huang, and X. Yu, “Object-goal visual navigation via effective exploration of relations among historical navigation states,” in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 2563–2573.
10. G. E. Box and G. M. Jenkins, “Time series analysis,” Forecasting and Control, 1970.

11. R. F. Engle, “Autoregressive conditional heteroscedasticity with estimates of the variance of united kingdom inflation,” Econometrica: Journal of the econometric society, pp. 987–1007, 1982.
12. D. Salinas, V. Flunkert, J. Gasthaus, and T. Januschowski, “Deepar: Probabilis- tic forecasting with autoregressive recurrent networks,” International Journal of Forecasting, vol. 36, no. 3, pp. 1181–1191, 2020.
13. Y. Jia, Y. Lin, X. Hao, Y. Lin, S. Guo, and H. Wan, “Witran: Water-wave infor- mation transmission and recurrent acceleration network for long-range time series forecasting,” Advances in Neural Information Processing Systems, vol. 36, 2024.
14. M. Liu, A. Zeng, M. Chen, Z. Xu, Q. Lai, L. Ma, and Q. Xu, “Scinet: Time series modeling and forecasting with sample convolution and interaction,” Advances in Neural Information Processing Systems, vol. 35, pp. 5816–5828, 2022.
15. H. Wu, T. Hu, Y. Liu, H. Zhou, J. Wang, and M. Long, “Timesnet: Temporal 2d- variation modeling for general time series analysis,” in The eleventh international conference on learning representations, 2023.
16. D. Luo and X. Wang, “Moderntcn: A modern pure convolution structure for gen- eral time series analysis,” in The Twelfth International Conference on Learning Representations, 2024.
17. H. Wu, J. Xu, J. Wang, and M. Long, “Autoformer: Decomposition transform- ers with auto-correlation for long-term series forecasting,” Advances in Neural Information Processing Systems, vol. 34, pp. 22 419–22 430, 2021.
18. T. Zhou, Z. Ma, Q. Wen, X. Wang, L. Sun, and R. Jin, “Fedformer: Frequency en- hanced decomposed transformer for long-term series forecasting,” in International Conference on Machine Learning. PMLR, 2022, pp. 27 268–27 286.
19. Y. Nie, N. H. Nguyen, P. Sinthong, and J. Kalagnanam, “A time series is worth 64 words: Long-term forecasting with transformers,” arXiv preprint arXiv:2211.14730, 2022.
20. Y. Liu, T. Hu, H. Zhang, H. Wu, S. Wang, L. Ma, and M. Long, “itransformer: Inverted transformers are effective for time series forecasting,” in The Twelfth International Conference on Learning Representations, 2024.
21. S. Wang, H. Wu, X. Shi, T. Hu, H. Luo, L. Ma, J. Y. Zhang, and J. ZHOU, “Timemixer: Decomposable multiscale mixing for time series forecasting,” in International Conference on Learning Representations (ICLR), 2024.
22. Z. Xu, A. Zeng, and Q. Xu, “Fits: Modeling time series with 10k parameters,” in The Twelfth International Conference on Learning Representations, 2024.
23. X. Ma, X. Li, L. Fang, T. Zhao, and C. Zhang, “U-mixer: An unet-mixer architec- ture with stationarity correction for time series forecasting,” in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 38, no. 13, 2024, pp. 14 255–14 262.
24. A. Zeng, M. Chen, L. Zhang, and Q. Xu, “Are transformers effective for time series forecasting?” in Proceedings of the AAAI conference on artificial intelligence, vol. 37, no. 9, 2023, pp. 11 121–11 128. 18 J. Li et al.
25. Z. Li, S. Qi, Y. Li, and Z. Xu, “Revisiting long-term time series forecasting: An investigation on linear mapping,” arXiv preprint arXiv:2305.10721, 2023.
26. S. Lin, W. Lin, W. Wu, H. Chen, and J. Yang, “Sparsetsf: Modeling long-term time series forecasting with 1k parameters,” in Forty-first International Conference on Machine Learning, 2024.
27. J. Kaplan, S. McCandlish, T. Henighan, T. B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, and D. Amodei, “Scaling laws for neural language models,” arXiv preprint arXiv:2001.08361, 2020.

28. H. Du, X. Yu, and L. Zheng, "Vtnet: Visual transformer network for object goal navigation," arXiv preprint arXiv:2105.09447, 2021.
29. H. Du, X. Yu, F. Hussain, M. A. Armin, L. Petersson, and W. Li, "Weakly-supervised point cloud instance segmentation with geometric priors," in Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, 2023, pp. 4271–4280.
30. Q. Wen, L. Sun, F. Yang, X. Song, J. Gao, X. Wang, and H. Xu, "Time series data augmentation for deep learning: A survey," arXiv preprint arXiv:2002.12478, 2020.
31. H. Qin, L. Su, C. Jiang, C. Zhang, G. Wu, and Y. Zhang, "Time series data augmentation algorithm combining deep metric learning and variational encoder," in 2023 IEEE/IAS Industrial and Commercial Power System Asia (I&CPS Asia). IEEE, 2023, pp. 1218–1223.
32. Z. Cai, W. Ma, X. Wang, H. Wang, and Z. Feng, "The performance analysis of time series data augmentation technology for small sample communication device recognition," IEEE Transactions on Reliability, vol. 72, no. 2, pp. 574–585, 2022.
33. Y. Gao, C. A. Ellis, V. D. Calhoun, and R. L. Miller, "Improving age prediction: Utilizing lstm-based dynamic forecasting for data augmentation in multivariate time series analysis," in 2024 IEEE Southwest Symposium on Image Analysis and Interpretation (SSIAI). IEEE, 2024, pp. 125–128.
34. Z. Cui, W. Chen, and Y. Chen, "Multi-scale convolutional neural networks for time series classification," arXiv preprint arXiv:1603.06995, 2016.
35. A. Le Guennec, S. Malinowski, and R. Tavenard, "Data augmentation for time se- ries classification using convolutional neural networks," in ECML/PKDD workshop on advanced analytics and learning on temporal data, 2016.
36. T. Wen and R. Keyes, "Time series anomaly detection using convolutional neural networks and transfer learning," arXiv preprint arXiv:1905.13628, 2019.
37. K. Pearson, "Notes on regression and inheritance in the case of two parents," proceedings of the royal society of London, vol. 58, p. 240–242, 1895.
38. T. Kim, J. Kim, Y. Tae, C. Park, J.-H. Choi, and J. Choo, "Reversible instance normalization for accurate time-series forecasting against distribution shift," in International Conference on Learning Representations, 2021.
39. Y. Zhang and J. Yan, "Crossformer: Transformer utilizing cross-dimension de- pendency for multivariate time series forecasting," in The eleventh international conference on learning representations, 2023.
40. A. Das, W. Kong, A. Leach, S. K. Mathur, R. Sen, and R. Yu, "Long-term fore- casting with tide: Time-series dense encoder," Transactions on Machine Learning Research, 2023.
41. A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga et al., "Pytorch: An imperative style, high-performance deep learning library," Advances in neural information processing systems, vol. 32, 2019.
42. D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in International conference on machine learning, 2015