# IMA: An Imputation-based Mixup Augmentation Using Self-Supervised Learning for Time Series Data

**Nha Dang Nguyen[1,], Dang Hai Nguyen[2], Khoa Tho Anh Nguyen[3*]**

[1]Vietnam - Korea University of Information and Communication Technology, Da Nang City, Vietnam
[2]VNU University of Engineering and Technology, Ha Noi City, Vietnam
[3]Vietnamese – German University, Ho Chi Minh City, Vietnam
nhand.21it@vku.udn.vn, 24025015@vnu.edu.vn, 30421001@student.vgu.edu.vn

## Abstract

Data augmentation plays a crucial role in enhancing model performance across various AI fields by introducing variability while maintaining the underlying temporal patterns. However, in the context of long sequence time series data, where maintaining temporal consistency is critical, there are fewer augmentation strategies compared to fields such as image or text, with advanced techniques like Mixup rarely being used. In this work, we propose a new approach, Imputation-based Mixup Augmentation (IMA), which combines Imputed-data Augmentation with Mixup Augmentation to bolster model generalization and improve forecasting performance. We evaluate the effectiveness of this method across several forecasting models, including DLinear (MLP), TimesNet (CNN), and iTrainformer (Transformer), these models represent some of the most recent advances in long sequence time series forecasting. Our experiments, conducted on three datasets (ETT-small, Illness, Exchange Rate) from various domains and compared against eight other augmentation techniques, demonstrate that IMA consistently enhances performance, achieving 22 improvements out of 24 instances, with 10 of those being the best performances, particularly with iTrainformer imputation in ETT dataset. The GitHub repository is available at: https://github.com/dangnha/IMA.

## Introduction

Time series forecasting has gained significant attention, especially long sequence time-series forecasting (LSTF), due to its wide range of applications across domains such as finance, healthcare, energy management, meteorology, and urban planning (Chen et al. 2023b). Over the years, extensive research has focused on developing models and methodologies to predict future values based on historical data. Early statistical methods like Autoregressive Integrated Moving Average (ARIMA) and exponential smoothing were instrumental in shaping the field but struggled to handle complex temporal dependencies. The emergence of deep learning brought forth models like Recurrent Neural Networks (RNNs) (Elman 1990) and Long Short-Term Memory (LSTMs) (Hochreiter and Schmidhuber 1997), which excelled in capturing long-range dependencies in sequential data. Building on these advancements, frameworks such as Multi-Layer Perceptrons (MLPs) (Lai et al. 2018; Salinas et al. 2020; Oreshkin et al. 2020; Zeng et al. 2023), Convolutional Neural Networks (CNNs) (Huang et al. 2019; Zhou et al. 2021b; Li et al. 2021; LIU et al. 2022; Wu et al. 2023), and Transformer-based architectures (Wu et al. 2021; Zhang and Yan 2023; Tang and Cai 2023; Liu et al. 2024) have demonstrated state-of-the-art performance by improving temporal dependency modeling and scalability for diverse LSTF tasks throughout recent year (showed in figure 1). These advancements have created robust pipelines, emphasizing the importance of data preprocessing, feature extraction, and model optimization.

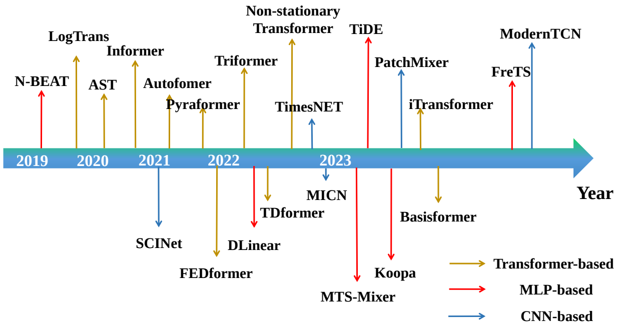

Figure 1: Key Milestones in Time Series Forecasting: Evolution of MLP-based, CNN-based, and Transformer-based Models (Liu and Wang 2024)

Despite the remarkable progress in long-term time series forecasting, a significant challenge is the lack of effective data augmentation techniques for time series data (Wen et al. 2021). In contrast to augmentation in fields such as Computer Vision (CV) and Natural Language Processing (NLP) (Pellicer, Ferreira, and Costa 2023), where data augmentation has become a cornerstone for improving model robustness and generalization, time series remains underexplored in this regard. In CV, augmentation techniques like flipping, cropping, jitter, GridMask, CutMix, and Mixup, among others, have been greatly explored (Xu et al. 2023). Similarly, some techniques in NLP, like prompt-based augmentation and masked language model (MLM)-driven data generation, have further expanded the toolkit for NLP data augmenta-

[*]This research is supported by AI VIETNAM

tion. In time series research, existing approaches, such as jittering, scaling, and time warping, provide some benefits but lack the sophistication and diversity seen in CV and NLP. Moreover, these methods often fail to capture the complex temporal patterns inherent in time series data (Chen et al. 2023a). Recent research has begun to explore latent augmentation methods like Mixup, which interpolates between data samples to create new, plausible instances, offering a promising direction for time series augmentation. Nevertheless, the field still demands more innovative techniques to address the unique challenges of time series data.

Another critical aspect of time series data is imputation, traditionally used to fill in missing data points in incomplete datasets. While imputation methods like interpolation, K-nearest neighbors (KNN), and deep learning-based approaches have proven effective for handling missing values, their potential as augmentation tools remains largely untapped. Existing imputation techniques focus solely on data recovery rather than leveraging imputation models to enhance data diversity. In this paper, we address these challenges by introducing an intensive augmentation method called Imputation-based Mixup Augmentation (IMA).

- We propose Imputated-data Augmentation with Self-Supervised Reconstruction (SSL), which takes advantage of imputation, capturing underlying data patterns and enriching the data diversity.
- We propose Imputation-based Mixup Augmentation (IMA), combining Imputed-data Augmentation with the Mixup Augmentation technique to enhance the model's ability to generalize and performance.
- We assess the performance of our method across three representative time series forecasting models—DLinear (MLP-based) (Zeng et al. 2023), TimesNet (Wu et al. 2023) (CNN-based), and iTransformer (Liu et al. 2024) (Transformer-based)—demonstrating its effectiveness in improving forecasting performance. Through these experiments, we highlight how our proposed method enhances the data diversity, enabling models to better generalize across various forecasting scenarios.

## Related Work

In recent years, **long sequence time-series forecasting (LSTF)** has become a focal point of research due to its critical applications in various domains. The advent of Transformer-based models has been particularly transformative, offering robust solutions for capturing long-term dependencies. Models such as Informer (Zhou et al. 2021a) and Autoformer (Wu et al. 2021) leverage sparse attention mechanisms and series decomposition to reduce computational overhead, while FEDformer (Zhou et al. 2022) introduces frequency-domain techniques like Fourier and wavelet transformations to enhance predictions of periodic data. Concurrently, CNN-based approaches such as TCN (Bharilya and Kumar 2024) and SCINet (LIU et al. 2022) have exploited dilated convolutions and hierarchical downsampling to efficiently capture both local and global temporal patterns, albeit with limitations in handling complex long-term dependencies. RNNs (Elman 1990), including variants like LSTM (Hochreiter and Schmidhuber 1997) and GRU (Cho et al. 2014), have been widely adopted for their sequential modeling capabilities, with enhancements such as attention mechanisms in DA-RNN (Qin et al. 2017) and hybrid architectures like ES-LSTM (Smyl 2020) improving their effectiveness in multivariate LSTF tasks. Lastly, as shown in figure 1, MLP-based methods, although traditionally less effective for sequential data, have seen renewed interest through feature-engineered adaptations, providing lightweight alternatives for simpler forecasting scenarios. These diverse approaches underscore the ongoing advancements in balancing performance, scalability, and efficiency for LSTF, driven by the unique challenges of multivariate and long-horizon forecasting tasks.

**Data augmentation** for time series data has gained significant attention as a means of enhancing model performance, particularly in tasks constrained by limited labeled data. Traditional methods such as time-domain transformations, including window cropping, slicing, and warping, are widely used due to their simplicity and effectiveness in introducing variability into datasets (Wen et al. 2021). More advanced approaches, such as decomposition-based methods (STL (Ouyang, Ravier, and Jabloun 2021) or Robust-STL (Wen et al. 2019) ) and generative models like GANs and VAEs, further enrich the augmentation landscape by maintaining the structural integrity of time series data while creating diverse synthetic samples. Despite these advancements, mixup — a technique that blends two samples to create new ones, remains underexplored in the context of time series, with few studies investigating its potential (Zhou et al. 2023), highlighting a critical gap in current research.

Furthermore, **imputation**, an essential tool for reconstructing missing time series data, has developed into a robust field with numerous methods. Techniques range from statistical methods like mean or linear interpolation to machine learning approaches such as k-nearest Neighbors (kNN), Gaussian Processes (Jafrasteh et al. 2023), and deep learning models like RNNs, GRUs, and Transformers (Wang et al. 2024a). These methods not only restore data but also capture underlying temporal dependencies, making them valuable for data augmentation. However, explicitly leveraging imputation models to augment time series data has yet to be thoroughly investigated, motivating us to propose IMA, which is introduced in the following section.

## Methodology

In this work, our approach is divided into two phases. In the first phase, Self-Supervised Reconstruction (SSR), we employ an imputation model that reconstructs masked input data to effectively capture underlying data patterns. This pretraining step enables the model to learn intrinsic structures in the time series data. In the second phase, Imputed-data Augmentation (IA) and Imputation-based Mixup Augmentation (IMA), we leverage the pre-trained imputation model for Imputed-data Augmentation to enhance data diversity, then integrate it with Mixup Augmentation (IMA). This combination enriches the variability in data representations by utilizing Mixup's ability to blend samples, thereby enhancing model performance across diverse time series scenarios.

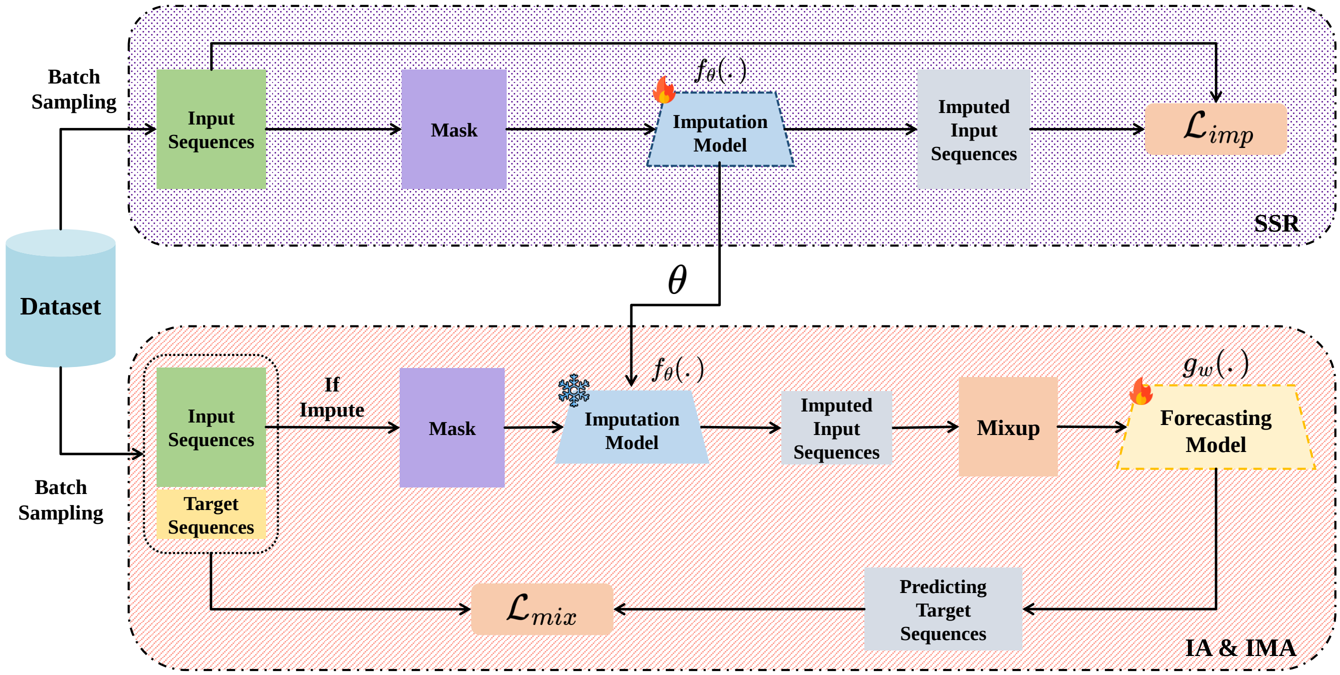

Figure 2: Illustration of the proposed data augmentation framework, comprising two key phases: Self-Supervised Reconstruction (SSR) for learning intrinsic data patterns and Imputed-based Mixup Augmentation (IMA) for enhancing data diversity and model generalization.

## Annotation Definition

Let $\mathcal{B} = \{(\mathbf{X}^{(i)}, \mathbf{Y}^{(i)})\}_{i=1}^{|\mathcal{B}|}$ represent a batch of $|\mathcal{B}|$ samples randomly sampled from the original dataset $\mathcal{D}$, where

$$\begin{aligned} \mathcal{D} = \{(\mathbf{X}^{(i)}, \mathbf{Y}^{(i)}) \mid &\mathbf{X}^{(i)} \in \mathbb{R}^{T_\mathbf{X} \times N_\mathbf{X}}, \\ &\mathbf{Y}^{(i)} \in \mathbb{R}^{T_\mathbf{Y} \times N_\mathbf{Y}}, \\ &i \in [1, |\mathcal{D}|]\}. \end{aligned} \tag{1}$$

Here, $\mathbf{X}^{(i)}$ denotes the input sequence, and $\mathbf{Y}^{(i)}$ denotes the corresponding target sequence for sample $i$. The parameters $T_\mathbf{X}$ and $T_\mathbf{Y}$ represent the number of time steps in $\mathbf{X}$ and $\mathbf{Y}$, respectively, while $N_\mathbf{X}$ and $N_\mathbf{Y}$ indicate the number of features per time step for $\mathbf{X}$ and $\mathbf{Y}$.

In more detail, a batch $\mathcal{B}$ contains $|\mathcal{B}|$ samples, where each input sequence $\mathbf{X}^{(i)}$ is represented as

$$\mathbf{X}^{(i)} = [\mathbf{X}_1^{(i)}, \mathbf{X}_2^{(i)}, \ldots, \mathbf{X}_{T_\mathbf{X}}^{(i)}], \tag{2}$$

with $\mathbf{X}_t^{(i)} \in \mathbb{R}^{N_\mathbf{X}}$ for $t = 1, \ldots, T_\mathbf{X}$. Each time step $\mathbf{X}_t^{(i)}$ is defined as

$$\mathbf{X}_t^{(i)} = [x_1, x_2, \ldots, x_{N_\mathbf{X}}], \tag{3}$$

where $x_j$ represents the $j$-th feature at time step $t$ for $j = 1, \ldots, N_\mathbf{X}$.

## Self-Supervised Reconstruction (SSR)

Self-supervised learning has been shown to enhance downstream tasks by uncovering inherent patterns within the data itself. In this work, we focus on its application in imputation for time series data.

Initially, we apply masking $\mathcal{M}_{SSR} \in \mathbb{R}^{|\mathcal{B}| \times T_\mathbf{X} \times N_\mathbf{X}}$ to the input data. In detail, for each sample $\mathbf{X}^{(i)}$ in the a batch data $\mathcal{B}$, we generate a masked version $\mathbf{X}_m^{(i)}$ defined by:

$$\mathbf{X}_m^{(i)} = \{\mathbf{X}_t^{(i)} \odot \mathbf{M}_t^{(i)} \mid t = 1, 2, \ldots, T_\mathbf{X}\} \tag{4}$$

where $\mathbf{X}_t^{(i)} \in \mathbb{R}^{N_\mathbf{X}}$ represents the feature vector at time step $t$ for the $i$-th sample (as defined in (2)), and $\mathbf{M}_t^{(i)} \in \{0,1\}^{N_\mathbf{X}}$ (where $\mathbf{M}^{(i)} \in \mathbb{R}^{T_\mathbf{X} \times N_\mathbf{X}}$ = $\{\mathbf{M}_t^{(i)}\}_{t=1}^{T_\mathbf{X}}$) is the corresponding binary mask vector indicating the observation status of the respective feature in $\mathbf{X}_t^{(i)}$ (as illustrated in figure 3). The binary value of each element in $\mathbf{M}_t^{(i)}$ is determined by a predefined mask_rate. For each feature in the input, we generate random values between 0 and 1, sampled from a uniform distribution, and if the value is less than or equal to mask_rate, the corresponding element in the mask is set to 0 (indicating the feature is masked). Otherwise, the value is set to 1 (indicating the feature remains observed). After getting $\{\mathbf{X}_m^{(i)}\}_{i=1}^{|\mathcal{B}|}$, the objective is to utilize an imputation model $f_\theta$ (where $\theta$ denotes the model parameters) to reconstruct the original input data $\{\mathbf{X}^{(i)}\}_{i=1}^{|\mathcal{B}|}$ from a masked version $\{\mathbf{X}_m^{(i)}\}_{i=1}^{|\mathcal{B}|}$. The model processes the masked data as input and generates imputed data as output $\mathbf{X}_{imp}^{(i)} = f_\theta(\mathbf{X}_m^{(i)})$ (as showed in SSR phase of figure 2). Finally, to guide the im-

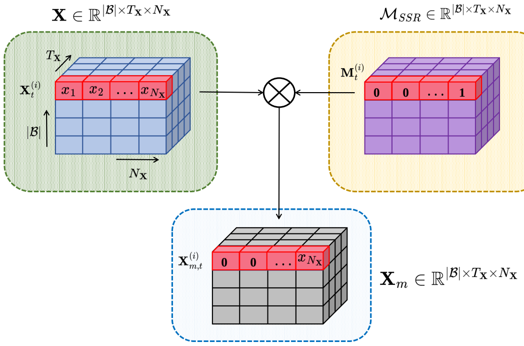
$\mathbf{X} \in \mathbb{R}^{|\mathcal{B}| \times T_{\mathbf{X}} \times N_{\mathbf{X}}}$
$\mathcal{M}_{SSR} \in \mathbb{R}^{|\mathcal{B}| \times T_{\mathbf{X}} \times N_{\mathbf{X}}}$
$T_{\mathbf{X}}$
$\mathbf{X}_t^{(i)}$
$x_1$
$x_2$
$\dots$
$x_{N_{\mathbf{X}}}$
$|\mathcal{B}|$
$N_{\mathbf{X}}$
$\mathbf{M}_t^{(i)}$
0
0
$\dots$
1
$\mathbf{X}_{m,t}^{(i)}$
0
0
$\dots$
$x_{N_{\mathbf{X}}}$
$\mathbf{X}_m \in \mathbb{R}^{|\mathcal{B}| \times T_{\mathbf{X}} \times N_{\mathbf{X}}}$

Figure 3: Illustration of the data masking strategy.

putation model, MSE loss between original input sequence and imputed input sequence was applied:

$$\mathcal{L}_{imp} \;=\; \frac{1}{|\mathcal{B}|} \sum_{i=1}^{|\mathcal{B}|} \sum_{t=1}^{T_{\mathbf{X}}} (1 - \mathbf{M}_t^{(i)}) \cdot \left( \mathbf{X}_t^{(i)} - \mathbf{X}_{imp,t}^{(i)} \right)^2 \quad (5)$$

## Imputed-data Augmentation (IA)

After sampling batches, We define a binary vector $\mathbf{i} \in \mathbb{R}^B$ where $B$ is the number of batches sampled from dataset $\mathcal{D}$. Each element $\mathbf{i}^{(i)}$ in the vector is determined by comparing a random number, drawn from a uniform distribution, with the imputation_rate. If the random number is smaller than imputation_rate, then $\mathbf{i}^{(i)} = 1$, indicating that the corresponding batch will undergo imputation-based augmentation. Otherwise, $\mathbf{i}^{(i)} = 0$, meaning no augmentation will be applied to the batch.

After the **Self-Supervised Reconstruction (SSR)** phase, we obtain a pre-trained model $f_\theta$ that captures the underlying temporal structures and patterns of the original data. This model is then used to reconstruct masked sequence $\mathbf{X}_m^{(i)}$ for each sample $\mathbf{X}^{(i)}$ in the batch $\mathcal{B}$. The masking is applied using a binary mask matrix $\mathcal{M}_{IMA} \in \mathbb{R}^{|\mathcal{B}| \times T_{\mathbf{X}} \times N_{\mathbf{X}}}$, resulting in the imputed sequence:

$$\mathbf{X}_{\text{imp}}^{(i)} = f_\theta(\mathbf{X}_m^{(i)}). \quad (6)$$

The reconstructed sequences are collected to form the augmented batch $\mathcal{B}_{\text{imp}}^{\mathbf{X}} = \{\mathbf{X}_{\text{imp}}^{(i)}\}_{i=1}^{|\mathcal{B}|}$. Next, these imputed sequences are passed into a forecasting model $g_w(\cdot)$, parameterized by $w$, to predict the future target sequences (illustrated in IA&IMA phase of figure 2 but without Mixup module). For each sample, the prediction is computed as:

$$\hat{\mathbf{Y}}^{(i)} = g_w(\mathbf{X}_{\text{imp}}^{(i)}). \quad (7)$$

The imputation process helps mitigate biases by filling in missing values with plausible estimates, thereby Imputed-data Augmentation introducing greater diversity to the data while preserving the underlying patterns.

## Imputed-based Mixup Augmentation

After obtaining the imputed batch $\mathcal{B}_{\text{imp}}^{\mathbf{X}}$, we apply Mixup Augmentation to generate additional synthetic data and enhance model generalization. Mixup interpolates between pairs of samples in the batch $\mathcal{B}_{\text{imp}}$ to create synthetic data points. For each batch $\mathcal{B}$, we draw a mixing coefficient $\lambda \sim \text{Beta}(\alpha, \alpha)$, where $\lambda \in [0, 1]$. This coefficient $\lambda$ governs the degree of mixing between the selected pairs of samples. As illustrated in the figure 4. Next, we randomly select two imputed samples $\mathbf{X}_{\text{imp}}^{(i)}$ and $\mathbf{X}_{\text{imp}}^{(j)}$ within the batch $\mathcal{B}$, and construct the mixed input $\mathbf{X}_{\text{mix}}^{(i,j)}$ as follows:

$$\mathbf{X}_{\text{mix}}^{(i,j)} = \lambda \cdot \mathbf{X}_{\text{imp}}^{(i)} + (1 - \lambda) \cdot \mathbf{X}_{\text{imp}}^{(j)} \quad (8)$$

The mixed sample $\mathbf{X}_{\text{mix}}^{(i,j)}$ is then passed to the forecasting model $g_w$ ($w$ denoted as model's parameters). For each mixed sample, the loss is computed as:

$$\mathcal{L}_{\text{mix}} = \lambda \cdot \mathcal{L}(g_w(\mathbf{X}_{\text{mix}}^{(i,j)}), \mathbf{Y}^{(i)}) + (1 - \lambda) \cdot \mathcal{L}(g_w(\mathbf{X}_{\text{mix}}^{(i,j)}), \mathbf{Y}^{(j)}) \quad (9)$$

where $\mathcal{L}$ denotes the forecasting loss, and $\mathbf{Y}^{(i)}$ and $\mathbf{Y}^{(j)}$ are the respective target sequences of the original samples $\mathbf{X}_{\text{imp}}^{(i)}$ and $\mathbf{X}_{\text{imp}}^{(j)}$. The calculation steps (in IA&IMA phase of figure 2) are shown in algorithm 1.

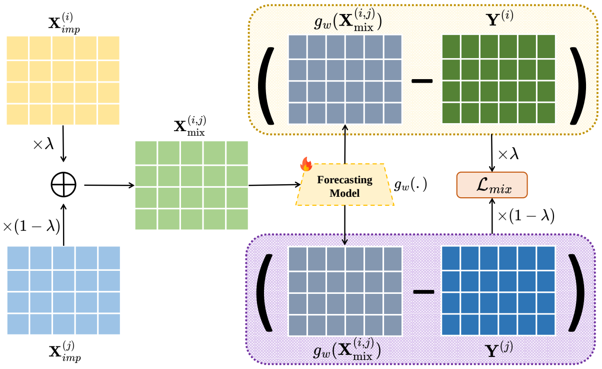

Figure 4: Mixup of two samples in imputed batch.

## Experiments and Results

**Dataset.** To assess the effectiveness of our data augmentation approach, we utilize three long-term time series forecasting datasets from diverse domains: ETT-small (energy), Illness (healthcare), and Exchange Rate (finance). **ETT small** (Zhou et al. 2021a) - a time series dataset recording the temperature of transformer stations for LTSF problem, which contains 2 separate datasets: ETTh for one-hour-level and ETTm for 15-minute-level, totaling 70,080 samples minutely level. Each data point consists of date, six features, and Oil Temperature target value, capturing both seasonal and irregular patterns. **Illness**[1] is a healthcare time series dataset that tracks weekly influenza-like illness

[1] Illness dataset: Weekly reported influenza-like illness (ILI) rates across regions in the United States. Data provided by the U.S. Centers for Disease Control and Prevention (CDC) through the FluView portal. Accessed December 1, 2024, gis.cdc.gov/grasp/fluview/fluportaldashboard.html.

Algorithm 1: Imputed-based Mixup Augmentation

```
Input: Batch $\mathcal{B} = \{(\mathbf{X}^{(i)}, \mathbf{Y}^{(i)})\}_{i=1}^{|\mathcal{B}|}$, imputation_rate, mask_rate
Define binary vector $\mathbf{i} \in \mathbb{R}^{B}$ and mask $\mathcal{M}_{IMA} \in \mathbb{R}^{|\mathcal{B}| \times T_{\mathbf{X}} \times N_{\mathbf{X}}}$
if $\mathbf{i}^{(i)} == 1$ then
    for each sample $\mathbf{X}_t^{(i)}$ in $\mathcal{B}$ do
        Compute masked input sequence $\mathbf{X}_m^{(i)}$ (refer to Formula 4)
        Compute imputed input sequence $\mathbf{X}_{\text{imp}}^{(i)}$ (refer to Formula 6)
    end for
    Randomly shuffle $\mathcal{B}_{imp}^{\mathbf{X}}$ to get $\mathcal{B}_{imp}'^{\mathbf{X}} = \{\mathbf{X}_{imp}^{(j)}\}_{j=1}^{|\mathcal{B}|}$
    Randomly select pairs $(\mathbf{X}_{\text{imp}}^{(i)}, \mathbf{X}_{\text{imp}}^{(j)})$ for $i, j \in [1, |\mathcal{B}|]$
    Compute mixed sample $\mathbf{X}_{\text{mix}}^{(i,j)}$ (refer to Formula 8)
    Compute loss for the mixed sample $\mathcal{L}_{\text{mix}}$ (refer to Formula 9)
    Update forecasting model's parameters:
    $w \leftarrow w - \text{lr} \cdot \nabla \mathcal{L}_{\text{mix}}$
else
    for each sample $\mathbf{X}_t^{(i)}$ in $\mathcal{B}$ do
        Pass input sequence into the forecasting model directly: $g_w(\mathbf{X}^{(i)})$
        Compute loss: $\mathcal{L}(g_w(\mathbf{X}^{(i)}), \mathbf{Y}^{(i)})$
        Update forecasting model's parameters:
        $w \leftarrow w - \text{lr} \cdot \nabla \mathcal{L}(g_w(\mathbf{X}^{(i)}), \mathbf{Y}^{(i)})$
    end for
end if
```

(ILI) rates across multiple regions. Each data point includes the date (in a weekly format), regional features such as population and healthcare capacity, and a target variable representing the ILI rate. And **Exchange Rate** (Lai et al. 2017), is a financial time series dataset that contains the daily exchange rates of eight foreign currencies, including the Australian Dollar, British Pound, Canadian Dollar, Swiss Franc, Chinese Yuan, Japanese Yen, New Zealand Dollar, and Singapore Dollar. The dataset spans the period from 1990 to 2016, with a total of 7,588 time steps at a daily sampling rate. Each data point includes the date and eight variables corresponding to the exchange rates of the respective currencies, making it a valuable resource for studying long-term forecasting and cross-variable correlations in financial markets.

**Experimental Setting.** We conducted experiments using the TSLib framework (Wu et al. 2023; Wang et al. 2024b), with three baseline models - DLinear, TimesNet, and iTransformer for long-term time series forecasting. These models represent recent advancements in the field and demonstrate strong performance across three distinct categories of time series modeling approaches. We applied seven common data augmentation methods (Jitter, Horizontal Flip, Vertical Flip, Scaling, Window Warp, Window Slide, Permutation) and the Mixup technique. Our proposed IMA method utilized grid search to optimize imputation_rate (0.125 for TimesNet imputation and iTransformer imputation) and mask_rate (0.375 for TimesNet imputation, 0.125 for iTransformer imputation). Using DLinear model for imputation task relies on simplified assumptions that limit its ability to capture the complex patterns inherent in the dataset. It consistently underperformed in our preliminary tests, which informed our decision to exclude it from our experiments. Finally, Model performance was evaluated using Mean Squared Error (MSE) and Mean Absolute Error (MAE).

**Results.** Table 1 shows that the Imputed-data Augmentation (IA) method significantly enhances model performance, particularly on the ETT dataset, where it achieves improvements in 20 out of 24 instances across baseline models and imputation methods. Notably, IA performs exceptionally well with the iTransformer forecasting model, improving performance in all 8 cases (as shown in figure 5).When combining IA with Mixup (IMA), the method remains strong on the ETT dataset, maintaining its ability to improve performance in 22 out of 24 instances, including 10 best-case results, particularly when paired with the iTransformer model. For the Illness and Exchange Rate datasets, IMA also performs well and slightly outperforms Mixup alone.

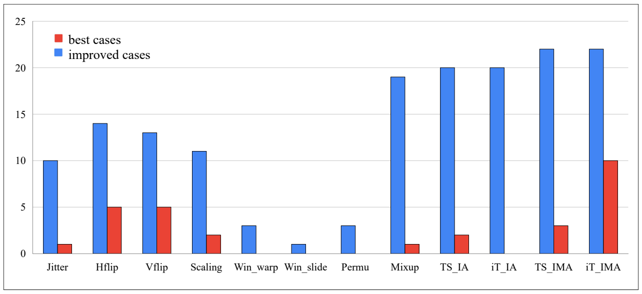

Figure 5: Comparison of the number of improvement cases and the best-case performance among eight augmentation methods, IA, and IMA on the ETT dataset.

However, IA and IMA or those augmentation methods struggle in specific scenarios, such as with the DLinear model on the ETTm1 dataset. This underperformance can be attributed to two primary reasons: the simplified architecture of the DLinear model, which limits its ability to capture complex temporal patterns, and the inherent complexity of the ETTm1 dataset, which demands models with more advanced temporal feature extraction capabilities. Furthermore, augmentation methods such as IA and IMA rely on generating diverse and informative data patterns. However, in the case of the DLinear model, its inability to leverage the augmented patterns effectively further exacerbates the performance gap.

For the Illness and Exchange Rate datasets, IA remains effective, achieving improvements in 4 out of 6 cases for both datasets. It stands out with peak performance in 2 cases for Illness and 4 cases for Exchange Rate (as shown in figure 6). Despite its overall success, certain challenges arise due to the characteristics of these datasets. Both the Illness and Exchange Rate datasets are relatively simple, leading to

| Model | DLinear | | TimesNet | | iTransformer | | DLinear | | TimesNet | | iTransformer | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Metric** | **MSE** | **MAE** | **MSE** | **MAE** | **MSE** | **MAE** | **MSE** | **MAE** | **MSE** | **MAE** | **MSE** | **MAE** |
| **Dataset** | **ETTh1** | | | | | | **ETTh2** | | | | | |
| Baseline | 0.445051 | 0.440448 | 0.458988 | 0.45493 | 0.447285 | 0.440212 | 0.479687 | 0.477797 | 0.406138 | 0.413908 | 0.4578 | 0.398759 |
| Jitter | 0.00E+00 | 1.00E-06 | -1.37E-03 | -7.73E-04 | -1.10E-05 | 7.06E-03 | 3.00E-05 | 2.20E-05 | 3.99E-03 | 3.63E-03 | -7.84E-02 | -1.13E-04 |
| Hflip | -5.73E-04 | -2.07E-04 | 7.94E-03 | 2.84E-03 | -5.47E-03 | -4.07E-03 | 1.24E-03 | 1.22E-03 | **-1.63E-02** | **-7.31E-03** | -5.92E-02 | **-2.66E-01** |
| Vflip | 4.41E-03 | 4.89E-03 | -6.30E-03 | -2.93E-03 | -5.86E-03 | -4.96E-03 | 7.55E-03 | 6.83E-03 | 3.36E-02 | 1.29E-02 | -7.43E-02 | -7.35E-04 |
| Scaling | 7.93E-04 | 1.14E-03 | -1.34E-03 | -6.42E-04 | **-2.98E-01** | **-2.93E-01** | -6.82E-04 | 1.97E-04 | 2.07E-02 | 1.29E-02 | -7.80E-02 | -2.83E-04 |
| Win_warp | 1.81E-02 | 1.81E-02 | 3.71E-02 | 2.06E-02 | 2.21E-02 | 1.30E-02 | -6.41E-04 | 9.71E-04 | -1.16E-04 | 3.98E-03 | -7.46E-02 | 2.40E-03 |
| Win_slide | 6.66E-02 | 4.17E-02 | 1.65E-02 | 1.50E-02 | 2.79E-02 | 1.76E-02 | 7.63E-03 | 7.55E-03 | 1.48E-02 | 1.23E-02 | -6.57E-02 | 9.67E-03 |
| Permu | -1.66E-04 | 4.87E-04 | -3.60E-03 | 1.44E-03 | 1.05E-02 | 1.76E-02 | 8.17E-03 | 6.42E-03 | 1.21E-02 | 5.43E-03 | -7.62E-02 | 1.34E-04 |
| Mixup | 2.10E-04 | 1.92E-04 | 7.78E-04 | -4.23E-04 | -1.58E-03 | -1.38E-03 | -1.01E-02 | -5.84E-03 | -2.67E-03 | -1.06E-03 | **-7.88E-02** | -3.87E-04 |
| TS_IA | -4.54E-03 | -3.23E-03 | -1.41E-04 | -1.48E-03 | -1.78E-03 | -1.89E-03 | -1.87E-02 | -1.20E-02 | 1.17E-03 | 1.02E-04 | -7.74E-02 | -2.00E-03 |
| iT_IA | -5.73E-03 | -4.84E-03 | -7.39E-03 | -7.65E-03 | -1.33E-03 | -1.96E-03 | -1.78E-02 | -1.16E-02 | -7.39E-03 | -3.62E-03 | -7.49E-02 | -3.09E-04 |
| TS_IMA | -6.31E-03 | **-6.11E-03** | -5.13E-03 | -6.76E-03 | -3.68E-03 | -3.60E-03 | **-2.15E-02** | **-1.30E-02** | -1.28E-02 | -7.06E-03 | -7.65E-02 | -1.84E-03 |
| iT_IMA | **-6.38E-03** | -5.85E-03 | **-1.54E-02** | **-1.18E-02** | -3.14E-03 | -3.98E-03 | -1.25E-02 | -8.32E-03 | -1.18E-02 | -6.75E-03 | -7.52E-02 | -2.02E-04 |
| **Dataset** | **ETTm1** | | | | | | **ETTm2** | | | | | |
| Baseline | 0.381687 | 0.390652 | 0.39177 | 0.403024 | 0.398893 | 0.394252 | 0.281894 | 0.358602 | 0.2544 | 0.307104 | 0.252632 | 0.312604 |
| Jitter | 1.50E-05 | **-8.95E-03** | 1.39E-04 | 4.12E-04 | -4.64E-03 | -4.60E-05 | 1.40E-05 | 1.60E-05 | 6.46E-03 | 3.53E-03 | -1.81E-04 | -2.21E-04 |
| Hflip | **-8.20E-05** | -2.56E-04 | 3.84E-03 | 2.34E-04 | -1.65E-02 | -2.11E-03 | 2.81E-03 | 3.15E-03 | -2.67E-03 | 1.20E-03 | **-4.76E-03** | -3.79E-03 |
| Vflip | 5.25E-04 | 1.00E-03 | -2.07E-03 | -4.10E-03 | -6.76E-03 | 3.11E-03 | -5.25E-03 | -2.63E-03 | -2.69E-03 | 1.28E-03 | 5.62E-02 | -3.79E-03 |
| Scaling | 3.79E-04 | 7.50E-04 | -2.01E-03 | -1.02E-03 | -1.80E-02 | 5.00E-05 | 2.25E-03 | 2.33E-03 | 7.54E-03 | 4.58E-03 | -4.20E-05 | 5.30E-05 |
| Win_warp | 7.35E-02 | 4.84E-02 | 4.79E-02 | 3.62E-02 | 4.49E-02 | 3.92E-02 | 3.46E-03 | 4.96E-03 | 3.92E-03 | 5.27E-03 | 2.42E-03 | 3.48E-03 |
| Win_slide | 8.62E-02 | 5.08E-02 | 8.04E-02 | 4.75E-02 | 3.45E-02 | 3.92E-02 | 1.50E-03 | 2.18E-03 | 1.33E-02 | 1.04E-02 | 4.73E-03 | 4.28E-03 |
| Permu | 4.45E-02 | 3.36E-02 | 7.88E-03 | 1.13E-02 | 3.45E-02 | 1.95E-02 | 5.75E-03 | 7.47E-03 | 1.33E-02 | 2.23E-03 | 1.93E-03 | 2.91E-03 |
| Mixup | 1.54E-04 | 4.21E-04 | -2.86E-03 | -2.19E-03 | -1.90E-02 | -2.12E-03 | -2.09E-03 | -1.48E-03 | -1.14E-03 | -9.32E-04 | -1.56E-03 | -1.10E-03 |
| TS_IA | 4.96E-03 | 4.27E-03 | -8.28E-03 | -2.50E-03 | -1.99E-02 | -1.15E-03 | **-2.18E-02** | **-2.18E-02** | -4.66E-03 | -3.04E-03 | -2.82E-03 | -3.71E-03 |
| iT_IA | 5.44E-03 | 4.72E-03 | -7.23E-03 | -3.52E-03 | -2.12E-02 | -1.20E-03 | 2.06E-03 | 1.87E-03 | -4.39E-03 | -2.77E-03 | -2.74E-03 | -3.71E-03 |
| TS_IMA | 5.18E-03 | 4.27E-03 | -7.62E-03 | -4.76E-03 | -2.17E-02 | -3.63E-03 | -1.49E-02 | -1.61E-02 | -5.80E-03 | -2.50E-03 | -4.03E-03 | -4.44E-03 |
| iT_IMA | 6.27E-03 | 5.27E-03 | **-1.22E-02** | **-9.31E-03** | **-2.36E-02** | **-4.22E-03** | -6.69E-03 | -5.99E-03 | **-7.18E-03** | **-4.58E-03** | -4.34E-03 | **-5.13E-03** |
| **Dataset** | **Illness** | | | | | | **Exchange Rate** | | | | | |
| Baseline | 4.003106 | 1.441318 | 1.998755 | 0.885458 | 1.807093 | 0.870089 | 0.168927 | 0.305395 | 0.219712 | 0.340417 | 0.180669 | 0.303503 |
| Jitter | 0.00E+00 | 0.00E+00 | -1.90E-05 | -3.00E-06 | 0.00E+00 | 0.00E+00 | 0.00E+00 | 0.00E+00 | -6.93E-04 | -3.37E-04 | 0.00E+00 | 0.00E+00 |
| Hflip | 0.00E+00 | 0.00E+00 | 3.30E-05 | 7.00E-06 | 0.00E+00 | 0.00E+00 | 0.00E+00 | 0.00E+00 | -3.72E-04 | -2.52E-04 | 0.00E+00 | 0.00E+00 |
| Vflip | 0.00E+00 | 0.00E+00 | -8.00E-06 | 3.00E-06 | 0.00E+00 | 0.00E+00 | 0.00E+00 | 0.00E+00 | -1.44E-03 | -1.29E-03 | 0.00E+00 | 0.00E+00 |
| Scaling | 0.00E+00 | 0.00E+00 | 1.10E-05 | 3.00E-06 | 0.00E+00 | 0.00E+00 | 0.00E+00 | 0.00E+00 | -1.14E-03 | -5.71E-04 | 0.00E+00 | 0.00E+00 |
| Win_warp | 0.00E+00 | 0.00E+00 | 1.90E-05 | 2.00E-06 | 0.00E+00 | 0.00E+00 | 0.00E+00 | 0.00E+00 | -1.90E-03 | -7.10E-04 | 0.00E+00 | 0.00E+00 |
| Win_slide | 0.00E+00 | 0.00E+00 | 1.30E-05 | 1.00E-06 | 0.00E+00 | 0.00E+00 | 0.00E+00 | 0.00E+00 | -8.52E-04 | -6.02E-04 | 0.00E+00 | 0.00E+00 |
| Permu | 0.00E+00 | 0.00E+00 | -2.30E-05 | -2.00E-06 | 0.00E+00 | 0.00E+00 | 0.00E+00 | 0.00E+00 | -3.70E-05 | 4.92E-04 | 0.00E+00 | 0.00E+00 |
| Mixup | 2.45E-02 | 4.30E-03 | -1.06E-01 | **-2.64E-02** | 3.93E-02 | 3.99E-03 | 7.51E-03 | 8.34E-03 | -8.73E-03 | -9.11E-03 | -1.08E-03 | -1.46E-03 |
| TS_IA | -9.15E-03 | -7.45E-04 | -7.46E-02 | -2.10E-02 | 3.21E-02 | 6.48E-03 | 4.18E-03 | 6.24E-03 | **-1.65E-02** | **-1.59E-02** | **-1.90E-03** | **-2.32E-03** |
| iT_IA | **-9.85E-03** | **-9.53E-04** | -7.03E-02 | -2.40E-02 | 4.09E-02 | 3.00E-03 | 7.16E-03 | 8.38E-03 | 4.83E-03 | 1.75E-03 | -1.46E-03 | -1.51E-03 |
| TS_IMA | -1.92E-03 | 2.27E-04 | -2.67E-03 | -1.23E-02 | 1.04E-01 | 2.12E-02 | 3.56E-03 | 5.04E-03 | -8.48E-03 | -9.61E-03 | -1.84E-03 | -2.12E-03 |
| iT_IMA | 1.60E-03 | 1.74E-03 | **-1.43E-01** | -2.54E-02 | 7.62E-02 | 1.13E-02 | 7.68E-03 | 1.00E-02 | -1.54E-02 | -1.48E-02 | -1.49E-03 | -1.89E-03 |

Table 1: Forecasting Performance Evaluation. Comparison of 8 augmentation methods with IM and IMA, using TimesNet (TS) and iTransformer (iT) for imputation-based enhancement.) **Red bold**: best case, **Blue**: improvement case, **Green Background**: Our methods.

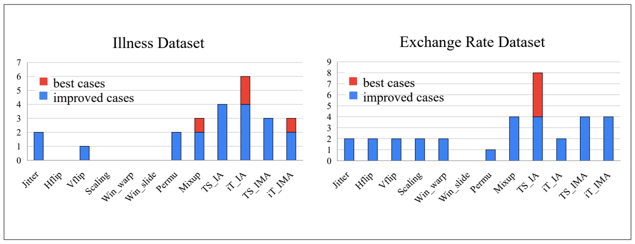

Figure 6: Comparison of the number of improvement cases and the best-case performance among eight augmentation methods, IA, and IMA on Illness, Exchange Rate datasets.

many zero values in the table. This simplicity allows models like DLinear and iTransformer to learn the patterns effectively without requiring significant augmentation. Consequently, the impact of augmentation methods is diminished in these cases, leading to limited improvements. In contrast, TimesNet, which relies on convolutional operation, exhibits greater sensitivity to the effects of augmentation. IA alone demonstrates greater effectiveness than IMA in these datasets, suggesting that standalone imputation may better address the data characteristics of Illness and Exchange Rate.

In conclusion, both IA and IMA provide strong and effective augmentation strategies, demonstrating consistent improvements across diverse models and datasets. While IA occasionally outperforms IMA on specific datasets, the combined approach of IMA offers a balanced and versatile solution compared to existing methods. Although IMA is not perfect in every scenario, it demonstrates improved effectiveness over existing methods by offering more uniform performance across diverse models and datasets.

## Conclusion

In this study, we propose Imputation-based Mixup Augmentation (IMA), a method that enhances time series forecasting by leveraging SSL training to capture trends and patterns in the data while preserving essential characteristics. By combining Imputation with Mixup, IMA not only increases data diversity but also improves model generalization, leading to better forecasting performance. Our results demonstrate that this approach outperforms Mixup alone, highlighting its potential to generate more diverse and resilient training data. Moreover, IMA may not yield optimal results for every forecasting model and dataset, but it opens promising avenues for further exploration and development in this direction.

## References

Bharilya, V.; and Kumar, N. 2024. Machine learning for autonomous vehicle's trajectory prediction: A comprehensive survey, challenges, and future research directions. *Vehicular Communications*, 46: 100733.

Chen, M.-H.; Xu, Z.; Zeng, A.; and Xu, Q. 2023a. FrAug: Frequency Domain Augmentation for Time Series Forecasting. *ArXiv*.

Chen, Z.; Ma, M.; Li, T.; Wang, H.; and Li, C. 2023b. Long sequence time-series forecasting with deep learning: A survey. *Information Fusion*, 97: 101819.

Cho, K.; van Merrienboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; and Bengio, Y. 2014. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. arXiv:1406.1078.

Elman, J. L. 1990. Finding structure in time. *Cognitive Science*, 14(2): 179–211.

Hochreiter, S.; and Schmidhuber, J. 1997. Long Short-Term Memory. *Neural Computation*, 9(8): 1735–1780.

Huang, S.; Wang, D.; Wu, X.; and Tang, A. 2019. DSANet: Dual Self-Attention Network for Multivariate Time Series Forecasting. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, CIKM '19, 2129–2132. New York, NY, USA: Association for Computing Machinery. ISBN 9781450369763.

Jafrasteh, B.; Hernández-Lobato, D.; Lubián-López, S. P.; and Benavente-Fernández, I. 2023. Gaussian processes for missing value imputation. *Knowledge-Based Systems*, 273: 110603.

Lai, G.; Chang, W.-C.; Yang, Y.; and Liu, H. 2017. Modeling Long- and Short-Term Temporal Patterns with Deep Neural Networks. *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*.

Lai, G.; Chang, W.-C.; Yang, Y.; and Liu, H. 2018. Modeling Long- and Short-Term Temporal Patterns with Deep Neural Networks. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, SIGIR '18, 95–104. New York, NY, USA: Association for Computing Machinery. ISBN 9781450356572.

Li, Y.; Li, K.; Chen, C.; Zhou, X.; Zeng, Z.; and Li, K. 2021. Modeling Temporal Patterns with Dilated Convolutions for Time-Series Forecasting. *ACM Trans. Knowl. Discov. Data*, 16(1).

LIU, M.; Zeng, A.; Chen, M.; Xu, Z.; LAI, Q.; Ma, L.; and Xu, Q. 2022. SCINet: Time Series Modeling and Forecasting with Sample Convolution and Interaction. In Oh, A. H.; Agarwal, A.; Belgrave, D.; and Cho, K., eds., *Advances in Neural Information Processing Systems*.

Liu, X.; and Wang, W. 2024. Deep Time Series Forecasting Models: A Comprehensive Survey. *Mathematics*, 12(10).

Liu, Y.; Hu, T.; Zhang, H.; Wu, H.; Wang, S.; Ma, L.; and Long, M. 2024. iTransformer: Inverted Transformers Are Effective for Time Series Forecasting. In *The Twelfth International Conference on Learning Representations*.

Oreshkin, B. N.; Carpov, D.; Chapados, N.; and Bengio, Y. 2020. N-BEATS: Neural basis expansion analysis for interpretable time series forecasting. In *International Conference on Learning Representations*.

Ouyang, Z.; Ravier, P.; and Jabloun, M. 2021. STL Decomposition of Time Series Can Benefit Forecasting Done by Statistical Methods but Not by Machine Learning Ones. *Engineering Proceedings*, 5(1).

Pellicer, L. F. A. O.; Ferreira, T. M.; and Costa, A. H. R. 2023. Data augmentation techniques in natural language processing. *Applied Soft Computing*, 132: 109803.

Qin, Y.; Song, D.; Cheng, H.; Cheng, W.; Jiang, G.; and Cottrell, G. W. 2017. A dual-stage attention-based recurrent neural network for time series prediction. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, IJCAI'17, 2627–2633. AAAI Press. ISBN 9780999241103.

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

Smyl, S. 2020. A hybrid method of exponential smoothing and recurrent neural networks for time series forecasting. *International Journal of Forecasting*, 36(1): 75–85. M4 Competition.

Tang, Y.; and Cai, H. 2023. Short-Term Power Load Forecasting Based on VMD-Pyraformer-Adan. *IEEE Access*, 11: 61958–61967.

Wang, J.; Du, W.; Cao, W.; Zhang, K.; Wang, W.; Liang, Y.; and Wen, Q. 2024a. Deep Learning for Multivariate Time Series Imputation: A Survey. arXiv:2402.04059.

Wang, Y.; Wu, H.; Dong, J.; Liu, Y.; Long, M.; and Wang, J. 2024b. Deep Time Series Models: A Comprehensive Survey and Benchmark. arXiv:2407.13278.

Wen, Q.; Gao, J.; Song, X.; Sun, L.; Xu, H.; and Zhu, S. 2019. RobustSTL: A Robust Seasonal-Trend Decomposition Algorithm for Long Time Series. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33: 5409–5416.

Wen, Q.; Sun, L.; Yang, F.; Song, X.; Gao, J.; Wang, X.; and Xu, H. 2021. Time Series Data Augmentation for Deep Learning: A Survey. In Zhou, Z.-H., ed., *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, 4653–4660. International Joint Conferences on Artificial Intelligence Organization. Survey Track.

Wu, H.; Hu, T.; Liu, Y.; Zhou, H.; Wang, J.; and Long, M. 2023. TimesNet: Temporal 2D-Variation Modeling for General Time Series Analysis. arXiv:2210.02186.

Wu, H.; Xu, J.; Wang, J.; and Long, M. 2021. Autoformer: Decomposition Transformers with Auto-Correlation for Long-Term Series Forecasting. In Beygelzimer, A.; Dauphin, Y.; Liang, P.; and Vaughan, J. W., eds., *Advances in Neural Information Processing Systems*.

Xu, M.; Yoon, S.; Fuentes, A.; and Park, D. S. 2023. A Comprehensive Survey of Image Augmentation Techniques for Deep Learning. *Pattern Recognition*, 137: 109347.

Zeng, A.; Chen, M.; Zhang, L.; and Xu, Q. 2023. Are transformers effective for time series forecasting? AAAI'23/IAAI'23/EAAI'23. AAAI Press. ISBN 978-1-57735-880-0.

Zhang, Y.; and Yan, J. 2023. Crossformer: Transformer Utilizing Cross-Dimension Dependency for Multivariate Time Series Forecasting. In *The Eleventh International Conference on Learning Representations*.

Zhou, H.; Zhang, S.; Peng, J.; Zhang, S.; Li, J.; Xiong, H.; and Zhang, W. 2021a. Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting. arXiv:2012.07436.

Zhou, T.; Ma, Z.; Wen, Q.; Wang, X.; Sun, L.; and Jin, R. 2022. FEDformer: Frequency Enhanced Decomposed Transformer for Long-term Series Forecasting. arXiv:2201.12740.

Zhou, Y.; Duan, Z.; Xu, H.; Feng, J.; Ren, A.; Wang, Y.; and Wang, X. 2021b. Parallel Extraction of Long-term Trends and Short-term Fluctuation Framework for Multivariate Time Series Forecasting. arXiv:2008.07730.

Zhou, Y.; You, L.; Zhu, W.; and Xu, P. 2023. Improving time series forecasting with mixup data augmentation. In *ECML PKDD 2023 International Workshop on Machine Learning for Irregular Time Series*.