Enhanced mixup for improved time series analysis

Khoa Tho Anh Nguyen [a,1], Khoa Nguyen [b,2], Taehong Kim [b,3], Ngoc Hong Tran [a,4], Vinh Quang Dinh [a,5,*]

[a] School of Electrical Engineering and Computer Science, Vietnamese–German University, Ho Chi Minh City 72000, Vietnam
[b] Information and Communication Technology, Chungbuk National University, Cheongju City, 28644, South Korea
[1] 30421001@student.vgu.edu.vn; [2] neko941@cbnu.ac.kr; [3] taehongkim@cbnu.ac.kr; [4] ngoc.th@vgu.edu.vn; [5] vinh.dinhquang@aivietnam.edu.vn
* corresponding author

ARTICLE INFO

Article history
Received May 11, 2024
Revised March 25, 2025
Accepted March 28, 2025
Available online May 31, 2025

Keywords
Time series augmentation
Mixed-sample augmentation
Time series forecasting
Time series classification

ABSTRACT

Time series data analysis is crucial for real-world applications. While deep learning has advanced in this field, it still faces challenges, such as limited or poor-quality data. In areas like computer vision, data augmentation has been widely used and highly effective in addressing similar issues. However, these techniques are not as commonly explored or applied in the time series domain. This paper addresses the gap by evaluating basic data augmentation techniques using MLP, CNN, and Transformer architectures, prioritized for their alignment with state-of-the-art trends in time series analysis rather than traditional RNN-based methods. The goal is to expand the use of data augmentation in time series analysis. The paper proposed EMixup, which adapts the Mixup method from image processing to time series data. This adaptation involves mixing samples while aiming to maintain the data's temporal structure and integrating target contributions into the loss function. Empirical studies show that EMixup improves the performance of time series models across various architectures (improving 23/24 forecasting cases and 12/24 classification cases). It demonstrates broad applicability and strong results in tasks like forecasting and classification, highlighting its potential utility across diverse time series applications.

1. Introduction

Time series analysis is vital for forecasting and classification in various sectors. Forecasting uses historical data to predict future outcomes, crucial for economic, stock market, and weather predictions. Classification identifies categories based on past data, which is important in medical diagnosis and financial pattern detection. This analysis helps in informed decision-making and trend anticipation. Deep learning has significantly improved time series forecasting and classification by effectively handling complex temporal data. Key models include Multi-Layer Perceptron (MLP), such as the LTSF-Linear [1], which are effective in Long-term time series forecasting (LTSF). Convolutional Neural Networks (CNN) like TimesNet [2] introduce novel 2D-variation modeling for enhanced adaptability. Recurrent Neural Networks (RNN) like TRUST [3], which is a new framework for trustworthy uncertainty propagation for time-series analysis. Transformers, exemplified by iTransformer [4], offer new ways to capture temporal dynamics while reducing complexity.

Deep learning's success in time series analysis often hinges on having large amounts of labeled data to prevent overfitting and ensure robustness, a challenging demand in real-world scenarios where data labeling is costly and time-consuming. In computer vision, data augmentation techniques like rotation and cropping expand datasets and improve model performance. However, applying similar strategies to

time series data, such as adding noise or warping, is more complex and needs to be explored. These methods must be carefully tailored to maintain crucial time series characteristics like trends and seasonality, highlighting the need for a nuanced understanding of specific data and tasks to ensure augmentations enhance rather than distort the data.

While time-series data augmentation was historically less focused than other modalities like image data, it is now receiving increasing attention [5]. This paper discusses time-series augmentation with basic methods involving value perturbation (jittering, scaling) and simple structure-based transformations like window slicing and sequence morphing (permutation). Advanced techniques include decomposition methods like STL, and populational generation using GANs and statistical models to create synthetic data.

Historically, time series data augmentation research has been skewed towards classification tasks, with less emphasis on forecasting. Moreover, basic augmentation techniques commonly applied in other domains have yet to be as thoroughly explored or implemented in time series analysis. Recognizing these gaps, this paper aims to extend the breadth of time series data augmentation research by addressing classification and forecasting tasks across various contemporary emerging deep learning. We prioritized CNN, MLP, and Transformer architectures over RNNs due to their alignment with SOTA time series trends (DLinear for simplicity, TimesNet for 2D periodicity, iTransformer for global dependencies), computational efficiency via parallelization and simple network (DLinear), and superior handling of long-term dependencies. Transformers' self-attention and CNNs' local pattern extraction address RNNs' sequential bottlenecks and memory constraints, ensuring scalability and accuracy in long-horizon tasks. In this study, we conduct a comprehensive experiment applying basic augmentation techniques to classification and forecasting tasks. In our study, we investigate the application of the Mixup augmentation strategy, traditionally used in image processing, to blend time series samples. Our goal is to preserve the inherent temporal structure of the time series as much as possible while integrating contributions from the targets into the loss function, making the strategy suitable for various time series tasks. Furthermore, this adaptation leads to the introduction of EMixup, an enhanced version of Mixup designed to further improve model generalization capabilities for time series data.

The main contributions of this paper are as follows:

- We empirically study basic time series data augmentation and Mixup techniques for forecasting tasks. This study evaluates the efficacy of these augmentations across a spectrum of state-of-the-art (SOTA) models, including MLP, CNN, and Transformer architectures.
- This research explores the application of basic time series data augmentation and Mixup strategies for classification challenges. We evaluate the performance of these techniques using various state-of-the-art models, including MLPs, CNNs, and Transformers.
- We propose the EMixup method tailored to meet the requirements of time series data in both classification and forecasting contexts. It improves baseline performance across various cases and frequently achieves the best results among basic augmentation methods

2. Related Work

2.1. Deep Learning Models for Time Series

Recent advancements in deep learning have significantly impacted the field of time series, with various models demonstrating remarkable performance across several benchmarks. Among these MLP-based models, the research [6] demonstrates the effectiveness of simple MLP architectures for financial time series forecasting. It finds that MLP can achieve competitive, accurate predictions and may even outperform more complex models like Transformers and RNNs in certain time series tasks. Other notable MLP-based approaches include LTSF-DLinear and LTSF-Nlinear [1], which contribute variations to traditional MLP structures to improve forecasting reliability. RNN-based models continue to excel in handling sequential data, with innovations like [7], which explore how multi-objective

evolutionary algorithms can enhance LSTM networks by optimally selecting key features, thus balancing complexity and forecasting accuracy for efficient time series analysis. Another variant [3] proposes a new framework (TRUST) for RNN architecture models, which introduces a Gaussian prior over network parameters and propagates the first two moments of the variational distribution to estimate uncertainty in time-series analysis. It also demonstrates robustness against noise and adversarial attacks. CNN-based models have also evolved to address time series tasks. [8] presents SIGAN-CNN, which leverages CNNs within a GAN for time series data augmentation, particularly beneficial for limited datasets. The generated data is then used with CNN-based classifiers to enhance the accuracy of time series classification. TimesNet [2] addresses the problem by modeling time series data as 2D-variation tensors, effectively capturing and analyzing complex temporal variations to achieve state-of-the-art results across various time series analysis tasks. Transformer-based models have brought significant innovations to the field. The TimesFM [9] is a pretrained Transformer utilizing a decoder-only structure and input patching for effective zero-shot time-series forecasting. The MultiPatchFormer [10] is a Transformer-based architecture employing multi-scale temporal patches and channel-wise attention to model complex dependencies in multivariate time-series forecasting. Moreover, the iTransformer [4] effectively utilizes attention and feed-forward networks on inverted dimensions of time series data to improve the accuracy and efficiency of time series forecasting.

2.2. Time Series Augmentation Methods

Basic augmentation techniques are essential for enhancing the robustness and performance of deep learning models when working with time series data. Techniques such as adding noise, permutation, scaling, and warping are essential for diversifying training scenarios, thereby helping models generalize better when encountering new data during training. Authors in [11] demonstrate that these techniques significantly enhance model training across various time series classification tasks. Additionally, in the realm of time series frequency components, [12] developed Random Noise Boxes (RNB), a novel technique for spectrogram classification, and [13] introduced Stratified Fourier Coefficients Combination (SFCC) to enhance time series datasets for deep neural networks. TF-FC explores the integration of both time and frequency domain techniques [14], which fuses time and frequency augmentations for enhanced time series representation. Advanced augmentation methods, including decomposition-based methods, statistical generative models, and learning methods, provide sophisticated approaches to enrich time series data. Decomposition-based methods isolate and augment different temporal components, such as trend and seasonality, which can be recombined to enhance model training. These methods have proven effective in complex real-world applications for forecasting, significantly improving predictive accuracy, as seen in the work by [15], [16]. Additionally, learning-based augmentation methods leverage deep learning models such as [17], [18] to generate new training samples, effectively improving model performance across diverse time series tasks. Mixup augmentation [19] has seen significant advancements since its introduction, with researchers proposing various extensions and improvements to address limitations like slow convergence, sensitivity to hyperparameters, and potential over-smoothing or under-representation of certain classes. Notable advancements include [20], Decoupled Mixup [21] for a better discrimination-smoothness trade-off, [22] for enhancing feature diversity and reducing over-reliance on filters, and GuidedMixup [23] that guides the mixing process. Additionally, Mixup has been successfully applied to various computer vision tasks beyond image classification [24], such as object detection [25] and robust image-denoising [26], demonstrating its versatility and effectiveness across diverse domains. Recent studies have explored adapting Mixup augmentation, initially proposed for image data, to time series data for classification and forecasting tasks. Methods like Mixup++ and LatentMixup++ [27] perform interpolations in the raw time series data and latent space of models, demonstrating improvements in classification tasks. An empirical study [28] also demonstrated that mix-based augmentations improve performance on physiological time series classification without requiring extensive tuning. For forecasting, techniques like TSMix [29] averaging standardized combinations of multiple actual univariate time series to generate augmented samples. These methods extend Mixup for time series by addressing challenges through novel interpolation strategies in time and latent spaces.

3. Method

3.1. Enhanced Mixup Technique

This research focuses on data augmentation techniques for time series analysis, particularly on forecasting and classification tasks. Fig. 1 shows an overview of the Enhanced Mixup Technique.

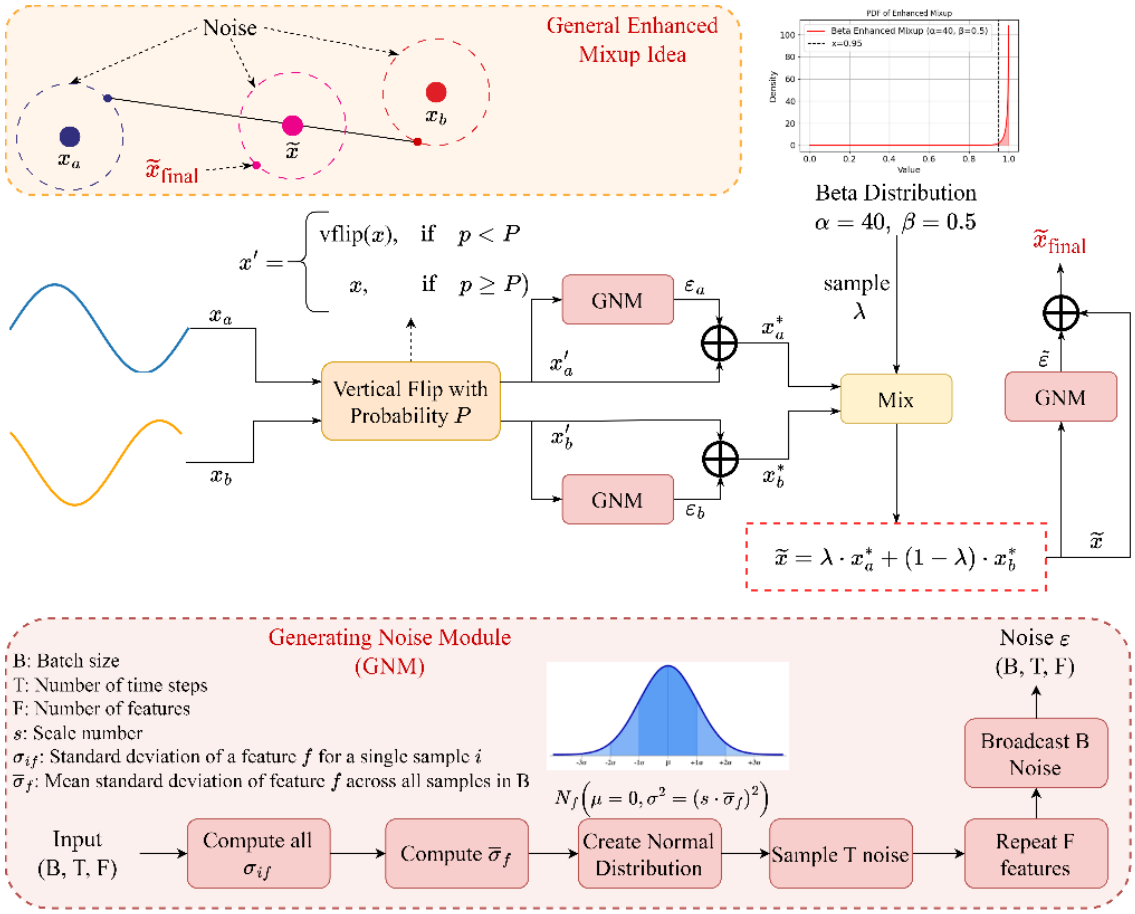

Fig. 1. Overview of the Enhanced Mixup Technique. Two time series, x_a and x_b are first modified by a GNM to add variability while maintaining structural integrity. They are then combined using Mixup to create $\tilde{x}$, further GNM processing to result in $\tilde{x}_{final}$. A vertical flip is occasionally applied to x_a and x_b with probability P to increase diversity

While data augmentation has thrived in other domains, such as computer vision, yielding significant improvements in model performance, there is a notable lack of comprehensive studies on augmentation techniques tailored explicitly for time series, especially in forecasting. To address this gap, we conduct a comprehensive empirical study, experimenting with various augmentation methods proposed in the past and adapting them for classification and forecasting tasks. Through detailed experiments, we demonstrate the limitations and instability of traditional augmentation techniques, which can sometimes lead to decreased model performance. As a more promising alternative, we propose a modifications Mixup augmentation, showcasing its stability and broad applicability across diverse time series datasets. Our experiments encompass recent SOTA models and emerging architectures in time series analysis, such as MLP, CNN, and Transformer-based models, going beyond the commonly studied RNN-based models. Our study spans 8 datasets, evaluating the impact of different augmentation techniques on model performance.

3.2. Mixup and Enhaced Mixup

Time series analysis involves input sequences $X=(x_1, x_2, x_3, ..., x_T)$ where x_t represents the value at time step t and T is the time series length, applicable in both univariate (x_t is scalar) and multivariate

(vector $x_t \in R^F$, where F is number of variables at time *t*) contexts. The outputs vary: forecasting aims to predict future values, outputting a sequence $\hat{Y} = (\hat{x}_{T+1}, \hat{x}_{T+2}, \hat{x}_{T+3}, ..., \hat{x}_{T+h})$ for a horizon *h*, whereas classification determines a class label for the sequence, resulting in an output $\hat{y}$ from a set of classes $\hat{y} \in \{1, 2, 3, ..., C\}$, often expressed as a probability distribution using a softmax function. Thus, while both tasks start from time-indexed sequences, they diverge in their predictive outputs, focusing on future data points in forecasting and class labels in classification.

Mixup is a data augmentation technique that generates synthetic training samples by combining pairs of samples: $\tilde{x} = \lambda x_a + (1-\lambda)x_b$ and labels: $\tilde{y} = \lambda y_a + (1-\lambda)y_b$, where $\lambda \sim \text{Beta}(\alpha,\alpha)$ controls the interpolation strength. Mixup enhances generalization by regularizing models through synthetic samples, but it risks disrupting the temporal structure in time series and may violate sequential dependencies or create unrealistic samples.

We introduce Enhanced Mixup (EMixup), an adaptation of the computer vision Mixup technique for time series, designed to preserve temporal structure while diversifying training data and avoiding degradation from basic augmentation methods. EMixup addresses the Mixup's inability to protect temporal structure in time series by using a highly skewed Beta distribution, which prioritizes one sample to minimize distortion. While this reduces sample diversity, techniques like the Generating Noise Module (GNM) are introduced to compensate, enhance diversity, and generate realistic synthetic data. The application of the loss function makes EMixup a robust adaptation for time series forecasting and classification tasks. Enhanced Mixup integrates Mixup into time series for forecasting and classification, ensuring simplicity and accessibility **as** a basic augmentation method. Fig. 2 shows a Comparison of Symmetric and Skewed Beta Distributions.

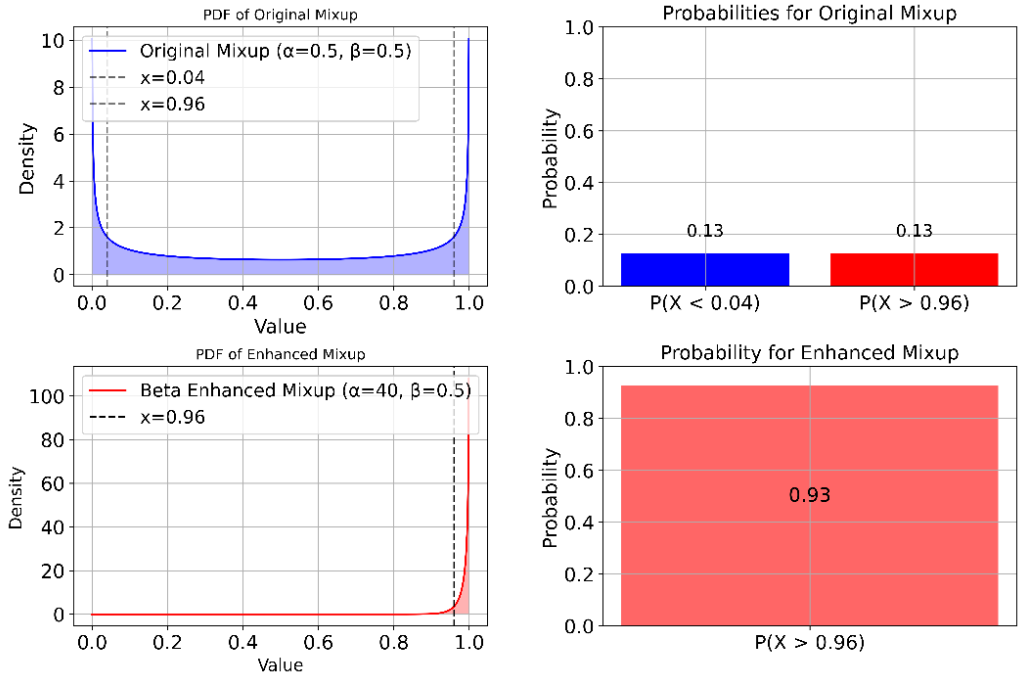

Fig. 2. Comparison of Symmetric and Skewed Beta Distributions. The top panel shows a symmetric Beta distribution (α=0.5, β=0.5), in contrast to the skewed distribution (α=40, β=0.5) in the bottom panel. In the skewed model, P(X>0.96)≈0.93 suggests a high probability of drawing values close to 1, while the symmetric model has probabilities P(X<0.04) and P(X>0.96) both around 0.13

The EMixup method, illustrated in Fig. 1, begins by processing two time series (x_a and x_b) through the GNM to add noise without significantly disrupting their structure. These noise-enhanced samples are then mixed using the Mixup technique (with highly skewed beta distribution) to create $\tilde{x}$, which undergoes further GNM processing to produce $\tilde{x}_{final}$. Additionally, a vertical flip is applied to (x_a or/and x_b) with probability P to enhance diversity. This process comprises five key steps:

- **Step 1.** During training, two time series samples (x_a and x_b) are selected and may undergo a vertical flip with probability P, producing x'_a and x'_b. This technique enhances model robustness by exposing it to reversed trends, broadening pattern recognition without additional data. Vertical flipping preserves temporal structure while altering patterns as trend directionality, aiding in learning invariant features. By using a small P, it diversifies training data while minimizing distributional changes and retaining essential series characteristics.
- **Step 2**. Noise variables ε_a and ε_b are added to the series x'_a and x'_b (vertically flipped or not), resulting in x^*_a and x^*_b. This is achieved using the GNM, which processes a batch of inputs with dimensions B (batch size), T (length of time steps), and F (number of features). The GNM generates noise for each feature *f* at every time step *t*, applying it uniformly across the entire batch B. Noise for each feature is drawn from a normal distribution N_f (μ=0, $\sigma^2=(s*\bar{\sigma}_f)^2$), where $\bar{\sigma}_f$ represents the mean standard deviation of the feature across the batch, and *s* is a scaling factor used to adjust the noise intensity. By leveraging the inherent variability of the training batch, the GNM ensures noise is proportional to natural fluctuations, preserving essential characteristics while preventing overfitting to overly smooth or noise-free data (see Fig. 3). This approach enhances model robustness by simulating real-world variations, diversifying data presentations, and subtly altering temporal dynamics without significantly disrupting the overall distribution.
- **Step 3.** We apply Mixup techniques to generate mixed samples $\tilde{x} = \lambda x^*_a + (1-\lambda)x^*_b$. To preserve the characteristics of x^*_a and prevent the destruction of the time series integrity, we modify the symmetric Beta distribution used to draw λ. Particularly, we employ a skewed Beta distribution, which favors values of λ close to 1.0. It minimizes the influence of x^*_b and maintains the primary structure of x^*_a. As shown in Fig. 2, the comparison between a symmetric Beta distribution Beta(α=0.5, β=0.5) and the skewed distribution Beta(α=40, β=0.5) shows a significant difference: while the probability P(X>0.96) in the skewed model is approximately 0.93, indicating strong preservation of x^*_a features, the P(X<0.04) and P(X>0.96) of the Beta(α=0.5, β=0.5) are almost 0.13. This method perverse the most original time series structure, which is critical for accurate analysis and prediction, while minimizing the risk of introducing unrealistic, distorted samples that could degrade model performance. However, prioritizing structure preservation may limit sample diversity. To address this, we propose that GNM enhance the diversity of generated samples. Overall, the skewed Beta distribution balances time series preservation with controlled variability.
- **Step 4.** The mixed samples $\tilde{x}$ then pass through the GNM to further diversify the mixed samples. This additional processing step enhances the robustness and variability of the dataset, ensuring that the Mixup-generated samples are not only blended but also introduce variations that enrich the training set. Refer to Fig. 3 for a comparison between EMixup and the original Mixup.
- **Step 5.** For the target, both forecasting and classification, we implement the mix operation in the loss function $L_{final} = \lambda L(\hat{y}, y_a)+(1-\lambda)L(\hat{y}, y_b)$, where $\hat{y}$ is the inference result from the model, y_a and y_b are the targets of the training sample. The loss function L can be cross-entropy loss for the classification, and L can be mean squared error for the forecasting task. This formulation captures the essential role of λ in determining the significance of each sample in the mixed data.

Fig. 3 illustrates the impact of different Mixup variations. The top-left shows a Mixup with λ = 0.5. From t = 0 to 2, the mixed sample follows series B downward while series A trends upward. From t = 2.5 to 3.2, the mixed sample follows series A downward, while series B trends upward. In this case, the mixed sample lacks consistency, depending on the dominant magnitude of series A and B at different time steps, and its behavior differs from A and B, resulting in an unrealistic sample. The top-right shows that λ = 0.95 (λ is very high, P(X>0.96) ≈ 0.93) increases the likelihood of preserving a mixed sample very similar to series A. The bottom-left demonstrates that adding noise from GNM distorts the sample slightly but still resembles series A. The bottom-right illustrates diversity achieved by applying a vertical flip, showing a result similar to A but inverted.

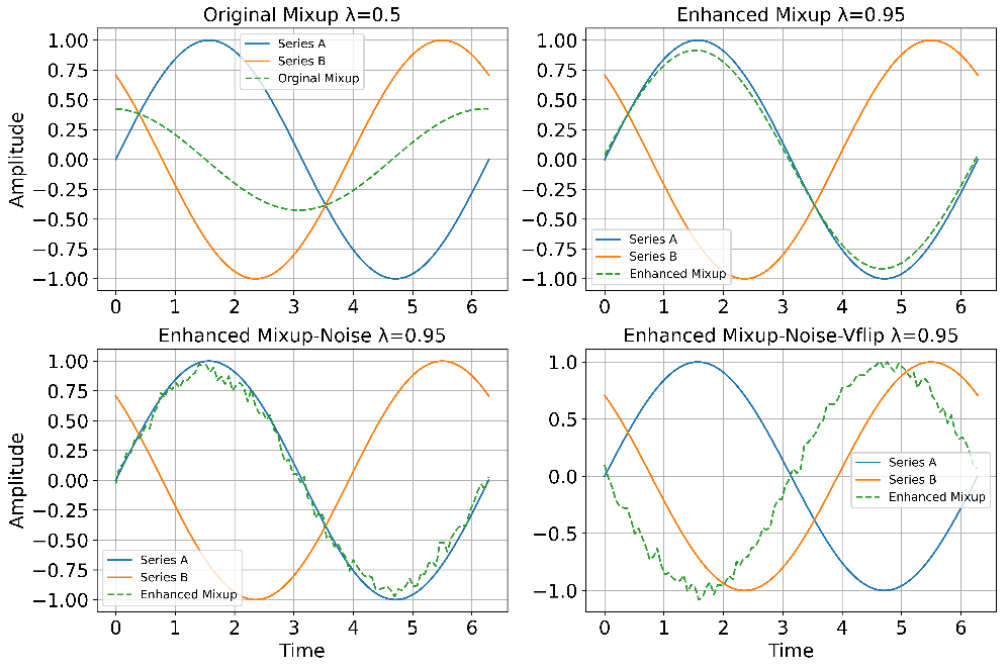

Fig. 3. Variations in Mixup with Different λ, Noise Addition, and Vertical Flip. Top left: λ=0.5 results in an even mix of two time series. Top right: λ=0.95 emphasizes features from one time series. Bottom left: Adds noise from a GNM to the λ=0.5 mix, enhancing variability. Bottom right: Incorporates a vertical flip before a λ=0.5 Mixup and noise addition, increasing diversity

EMixup distinguishes itself from a standard Mixup through several key adaptations for time series data. Unlike the symmetric Beta distribution in the standard Mixup, EMixup employs a highly skewed distribution, favoring the dominance of one original time series to preserve the temporal structure. It introduces GNM for controlled, feature-specific noise addition before and after mixing, enhancing robustness. EMixup also incorporates a vertical flip (with a small probability) for increased data diversity and modifies the loss function to integrate the contributions of targets based on the mixing ratio. These modifications tailor EMixup for effective application in both time series forecasting and classification tasks, unlike the primarily image-focused standard Mixup.

3.3. Experimental Setup

We implement test cases in these experiments following the Time Series Library (TSlib) [2]. There are key configurations: DLinear comprises two single-layer linear networks, each with neurons equal to the forecast horizon, modeling trend, and remainder components. This setup is consistent for both forecasting and classification. TimesNet analyzes the frequency domain, selecting the top-k (k = 5 for forecasting, task-specific for classification) significant frequencies. It uses an embedding layer, followed by two TimeBlocks [2] (each with two Inception Blocks containing six CNN layers), and uses a MLP layer for forecasting or classification. iTransformer uses two blocks for encoder and three blocks for classification. In the forecasting task, we use a 15% Vflip rate in EMixup, unlike the classification task which uses no Vflip. The noise scaling factor s is set to 0.01. These experiments evaluate time series augmentation by assessing improvements across models and datasets, and identifying augmentations with the most consistent performance gains.

In the domain of time series, various deep learning models have been tailored to leverage the unique characteristics of sequential data. Three recently notable models that have been developed are DLinear, TimesNet, and iTransformer. Each model corresponds to one of the base architectures in deep learning: MLP, CNN, and Transformer, respectively, showcasing different approaches to handle time-dependent patterns. DLinear is a time series forecasting model utilizing the MLP architecture to effectively learn relationships in time series through dense layers, adjusting traditional MLP structures to emphasize temporal dynamics. TimesNet, leveraging CNNs, introduces the TimesBlock module to transform 1D

time series into 2D, enabling complex pattern modeling through 2D convolutions and achieving state-of-the-art results in various forecasting tasks. iTransformer adapts the Transformer architecture for time series forecasting, using an inverted structure to treat time series as tokens and leveraging attention mechanisms and Transformer components to forecast high-dimensional multivariate series effectively.

In our time series forecasting experiments, we utilize four datasets [30]: ETTh1, ETTh2, ETTm1, and ETTm2, which consist of electricity transformer temperature data from two counties in China collected over two years to study the effects of data granularity on forecasting. ETTh1 and ETTh2 provide hourly data, while ETTm1 and ETTm2 offer minute-level granularity. We adopt a multivariate forecasting method, targeting multiple variables such as oil temperature and six power load features (see Table 1).

Table 1. Experiment Datasets for Time Series Forecasting and Classification. Forecasting uses 96 historical data points to predict 192 future steps, and classification involves predicting diverse classes

Forecasting Data					Classification Data			
Dataset	*Additional Features*	*Target Variable*	*Input Length*	*Prediction Length*	*Dataset*	*Input Length*	*Features*	*Num Classes*
ETTh1	6 features	Oil tempe.	96	192	Handwriting	152	3	26
ETTh2	6 features	Oil tempe.	96	192	Heartbeat	405	61	2
ETTm1	6 features	Oil tempe.	96	192	SelfRegulationSCP1	896	6	2
ETTm2	6 features	Oil temper.	96	192	SelfRegulationSCP2	1152	7	2

The datasets are split into 12 months for training, 4 months for validation, and 4 months for testing. In classification, we utilize four datasets from the UEA [31] time series classification repository: Handwriting, Heartbeat, SelfRegulationSCP1, and SelfRegulationSCP2. The Handwriting dataset contains three-dimensional accelerometer data from smartwatch-recorded handwriting motions. The Heartbeat dataset features heart sound recordings from both healthy and cardiac patients, analyzed using spectrograms. SelfRegulationSCP1 and SelfRegulationSCP2 provide EEG data from healthy subjects and an ALS patient (see Table 1).

4. Results and Discussion

These experiments aim to evaluate time series augmentation techniques from two angles: general enhancements across diverse models and scenarios and an analysis to pinpoint augmentations that consistently outperform others. Based on Fig. 4, it is clear that our proposed EMixup method generally excels in comparison to other methods. It demonstrates a remarkable capacity to improve performance broadly, topping the charts in the number of improved and best cases. In Fig. 4, improved cases indicate where an augmentation technique enhanced model performance, while best cases show a technique that outperformed all others to achieve the highest improvement.

The study assesses data augmentation techniques for time series forecasting using ETTh1, ETTh2, ETTm1, and ETTm2 datasets, employing DLinear (MLP-based), TimesNet (CNN-based), and iTransformer (Transformer-based) models. Each model's performance is evaluated using Mean Squared Error (MSE) and Mean Absolute Error (MAE), with results organized in Table 2 showing improvements over baseline values are underlined, and the most effective techniques are bolded. This approach allows for a detailed comparison of how different augmentation methods enhance forecasting accuracy across various sophisticated models. Overall, EMixup enhances performance across various models and datasets, improving the baseline in most test cases. The only exception is its performance on the ETTm2 dataset with the iTransformer model regarding MSE. Techniques such as window warping and window slicing fail to improve all datasets and models. Permutation exhibits similar behavior, except in the ETTh1 dataset. It is understood that these techniques may disrupt the temporal structure of time series, making them unsuitable for LTSF, which relies heavily on time step dependencies. Conversely, EMixup is designed to minimally impact and preserve the temporal structure as much as possible while generating diverse training data. Other techniques, such as horizontal flip, vertical flip, scaling, and jitter, also improve the performance but are still less effective than EMixup. For example, vertical flip may improve

outcomes on ETTm1 and ETTm2 but perform poorly on ETTh2. Despite having a minor probability of using vertical flip, EMixup consistently shows strong performance across all these datasets. Mixup results in improvements in 19 cases, similar to Horizontal flip in 17 cases, but Horizontal flip demonstrates superior outcomes to Mixup.

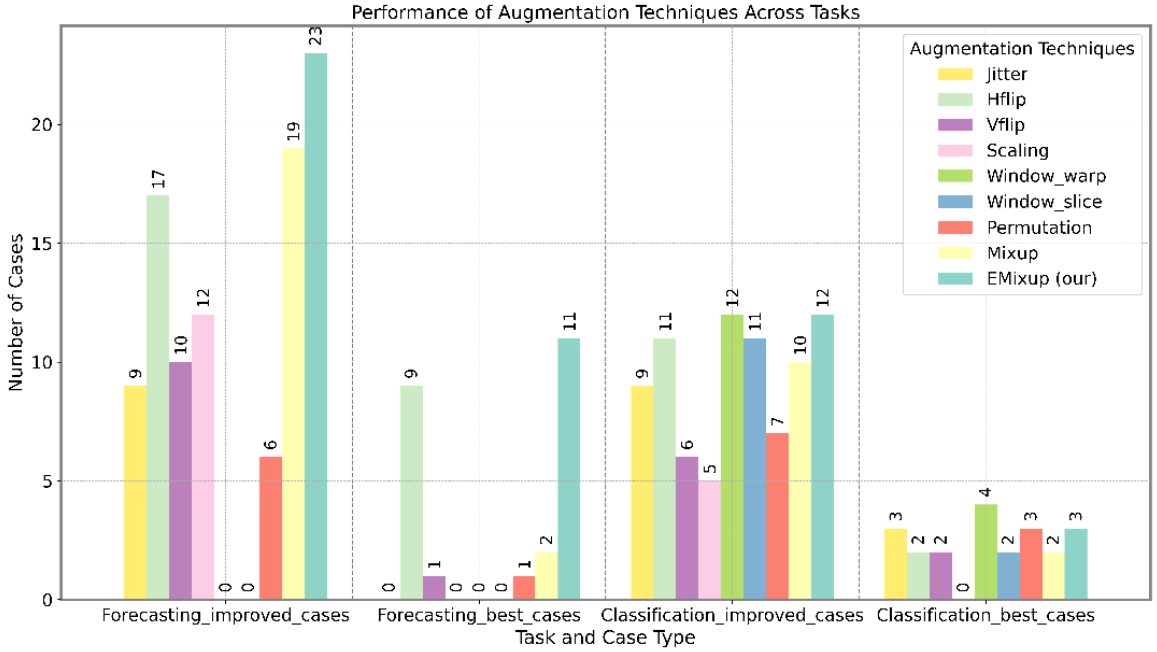

Fig. 4. Improvement of Different Augmentations on Model Performance. The bar chart shows the number of improved and best cases for each technique in forecasting and classification tasks. EMixup emerges as the majority winner, demonstrating its robustness across various datasets, models, and metrics

Furthermore, EMixup most frequently achieves superior performance, gaining the highest results 11 times compared to other techniques. Horizontal flip in second place, achieving top results 9 times, but not improving as many cases as EMixup (23 cases). This experiment shows that EMixup, based on the Mixup concept but modified for time series, aims to preserve the characteristics of time series data as much as possible while diversifying the training data (skewed Mixup ratio distribution, adding noise following the variance of the training batch, and minimal use of vertical flip). It is also noteworthy that EMixup remains active throughout training. In contrast, Mixup is activated only 50% of the time, as It can generate non-realistic samples, whereas our design maintains the integrity of the time series data.

The study employs a detailed comparative analysis of data augmentation techniques on time series classification using Handwriting, Heartbeat, SelfRegulationSCP1, and SelfRegulationSCP2 datasets.

We employ three advanced models: DLinear, TimesNet, and iTransformer, which were chosen for their distinct methods of processing time series data. We evaluate the effectiveness of each model and augmentation technique using Accuracy and F1-score, with the results detailed in Table 3. The most successful strategies are highlighted in bold, and improvements are underlined. This setup evaluates the effects of augmentation methods on model performance across various datasets, providing key insights into their effectiveness and adaptability. In classification, these augmentation strategies exhibit a distinct set of dynamics. Techniques such as vertical flipping and scaling are less effective, suggesting that their application might be too simplistic or disruptive for complex classification patterns. In contrast, permutation shows a slight improvement. These techniques also exhibit strong specificity to particular datasets; for instance, vertical flipping performs well on the SelfRegulationSCP2 dataset but poorly on the Handwriting and Heartbeat datasets. Other augmentations yield better results in terms of generalization and peak performance, with only minor differences among them, ranging from 10 to 12 improved cases and 3 to 4 best cases per augmentation.

Table 2. Time Series Forecasting Augmentation Techniques on Forecasting Dataset: Evaluation across DLinear (MLP), TimesNet (CNN), and iTransformer (Transformer) using MSE and MAE. Enhancements over baseline are underlined; most significant ones are bolded

	Model	DLinear		TimesNet		iTransformer	
	Metric	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	Baseline	0.444866	0.440218	0.439870	0.442437	0.448541	0.441158
	Jitter	1.40E-06	2.86E-06	1.66E-04	4.60E-04	-1.58E-06	-2.98E-07
	Hflip	-3.19E-04	1.81E-05	1.23E-03	-6.14E-04	**-6.71E-03**	-4.90E-03
	Vflip	3.66E-03	4.27E-03	1.00E-02	7.05E-03	-6.67E-03	**-5.64E-03**
	Scaling	9.82E-04	1.35E-03	-6.03E-04	-1.78E-04	-2.72E-04	-1.03E-04
	Window_warp	1.99E-02	1.51E-02	2.55E-02	1.81E-02	1.98E-02	1.10E-02
	Window_slice	6.78E-02	4.24E-02	3.63E-02	2.04E-02	2.71E-02	1.68E-02
	Permutation	-6.75E-05	6.58E-04	**-3.43E-03**	3.26E-03	-1.42E-03	-7.45E-04
	Mixup	9.02E-04	9.81E-04	-6.74E-04	**-1.23E-03**	-2.53E-03	-2.15E-03
	EMixup (our)	**-7.06E-04**	**-1.08E-03**	-6.06E-04	-9.51E-04	-1.27E-03	-1.69E-03
ETTh2	Baseline	0.481671	0.479034	0.396757	0.410084	0.381267	0.399462
	Jitter	2.71E-05	1.90E-05	1.32E-03	1.84E-03	-1.49E-04	-4.39E-05
	Hflip	-4.99E-04	4.75E-04	**-1.41E-02**	**-9.38E-03**	8.36E-04	-1.52E-03
	Vflip	1.14E-02	9.34E-03	3.19E-02	1.06E-02	2.57E-03	-1.37E-03
	Scaling	3.84E-03	2.50E-03	-3.00E-03	-8.36E-04	-6.89E-05	6.99E-05
	Window_warp	1.56E-03	2.47E-03	1.81E-04	3.62E-03	2.49E-03	1.89E-03
	Window_slice	7.87E-03	7.76E-03	1.23E-02	1.27E-02	1.30E-02	1.01E-02
	Permutation	1.40E-02	9.60E-03	4.93E-03	-8.80E-04	2.43E-03	5.62E-04
	Mixup	-1.06E-02	-5.54E-03	-5.06E-04	-1.13E-03	**-2.36E-03**	-1.91E-03
	EMixup (our)	**-2.46E-02**	**-1.28E-02**	-4.46E-03	-4.36E-03	-1.24E-03	**-2.16E-03**
ETTm1	Baseline	0.381856	0.391023	0.403705	0.410090	0.380991	0.395193
	Jitter	1.50E-05	9.83E-06	-6.65E-04	-1.11E-04	-7.03E-06	3.19E-05
	Hflip	-2.23E-04	**-9.23E-04**	-1.66E-02	**-1.34E-02**	4.15E-04	-3.72E-03
	Vflip	-4.36E-04	4.72E-05	-1.05E-02	-5.40E-03	8.86E-03	2.52E-03
	Scaling	4.30E-04	6.68E-04	-5.46E-04	1.87E-04	-2.43E-04	-1.82E-04
	Window_warp	7.26E-02	4.80E-02	3.82E-02	2.87E-02	6.29E-02	3.90E-02
	Window_slice	8.58E-02	5.05E-02	6.99E-02	3.85E-02	6.08E-02	3.77E-02
	Permutation	4.54E-02	3.41E-02	1.25E-03	7.82E-03	1.98E-02	1.73E-02
	Mixup	3.78E-05	6.32E-05	6.15E-03	-2.57E-04	-1.74E-03	-2.92E-03
	EMixup (our)	**-8.29E-04**	-7.96E-04	**-1.76E-02**	-1.26E-02	**-2.38E-03**	**-4.56E-03**
ETTm2	Baseline	0.284225	0.361074	0.258918	0.308863	0.253002	0.312666
	Jitter	2.72E-05	3.47E-05	1.99E-03	9.84E-04	-4.28E-05	-1.19E-04
	Hflip	4.50E-03	3.79E-03	**-9.89E-03**	**-3.49E-03**	**-5.44E-03**	**-4.20E-03**
	Vflip	1.67E-04	2.17E-03	-9.50E-03	-3.07E-03	-5.98E-04	-1.69E-03
	Scaling	3.25E-03	3.25E-03	4.50E-03	2.91E-03	-2.26E-04	-1.45E-04
	Window_warp	2.87E-03	4.16E-03	3.99E-03	5.23E-03	3.65E-03	4.32E-03
	Window_slice	6.45E-04	1.34E-03	1.30E-02	1.00E-02	6.10E-03	5.44E-03
	Permutation	6.94E-03	7.72E-03	7.07E-04	-1.81E-04	1.48E-03	2.06E-03
	Mixup	-4.39E-03	-2.97E-03	-4.20E-03	-2.56E-03	-6.14E-04	-6.55E-04
	EMixup (our)	**-2.49E-02**	**-2.35E-02**	-5.40E-03	-1.59E-03	1.14E-03	-1.06E-03

EMixup continues to perform well with the Handwriting and SelfRegulationSCP2 datasets, and despite showing no improvement in the SelfRegulationSCP1 dataset, it still achieves the second-best performance compared to other techniques. EMixup records 12 improved cases and 3 best cases, slightly less efficient than window warping, which has 12 improved cases and 4 best cases. The analysis of augmentation techniques across forecasting and classification tasks highlights distinct preferences and efficacies. Techniques such as EMixup demonstrate broad applicability and robust performance across both task types, suggesting their utility in diverse machine learning scenarios. On the other hand, task-specific variations are notable; for instance, window warping and window slice are more effective in

classification than forecasting, indicating their suitability for some specific tasks only. This highlights the need for strategically selecting augmentation techniques that enhance specific task characteristics, ensuring optimal model performance and generalization.

Table 3. Time Series Classification Augmentation Techniques on classification datasets: Evaluation across DLinear (MLP), TimesNet (CNN), and iTransformer (Transformer) using Accuracy and F1-score. Enhancements over baseline are underlined; most significant ones are bolded

	Model	DLinear		TimesNet		iTransformer	
	Metric	ACC	F1	ACC	F1	ACC	F1
Handwriting	Baseline	18.9412%	15.3887%	30.9412%	27.8433%	21.5294%	18.6035%
	Jitter	0.0000%	-0.0358%	0.2353%	0.4150%	-0.2353%	-0.2712%
	Hflip	-4.4706%	-4.9565%	-3.8824%	-4.5906%	-4.3529%	-5.5322%
	Vflip	-9.5294%	-8.2334%	-11.6471%	-12.3302%	-8.4706%	-9.4189%
	Scaling	0.2353%	0.2789%	-0.8235%	-0.2425%	0.0000%	0.0240%
	Window_warp	-4.8235%	-5.6240%	0.3529%	0.2820%	-3.6471%	5.3001%
	Window_slice	-1.5294%	-3.0908%	**2.0000%**	**1.3596%**	-3.2941%	-5.0463%
	Permutation	-3.1765%	-3.1572%	-3.4118%	-2.8838%	-1.7647%	-2.4659%
	Mixup	**0.3529%**	**0.3693%**	-0.5882%	-0.3561%	0.1176%	-0.6718%
	EMixup (our)	0.0000%	0.2080%	0.4706%	0.4057%	**1.5294%**	**1.1307%**
Heartbeat	Baseline	68.7805%	63.3356%	74.1463%	65.1863%	74.1463%	66.0734%
	Jitter	0.4878%	0.0791%	**2.4390%**	0.0875%	**2.4390%**	**1.2833%**
	Hflip	0.9756%	1.4528%	-0.9756%	0.0368%	-3.4146%	-2.5243%
	Vflip	-0.4878%	0.7722%	-1.4634%	-5.0474%	0.9756%	-0.5090%
	Scaling	0.9756%	0.1414%	-1.9512%	-3.1382%	0.0000%	0.0000%
	Window_warp	2.9268%	0.2099%	0.0000%	-5.2180%	-0.4878%	-0.8584%
	Window_slice	3.4146%	-0.7767%	-5.3659%	-4.0673%	0.0000%	0.0000%
	Permutation	**5.8537%**	**2.7327%**	-0.4878%	**0.8733%**	-2.4390%	-0.9373%
	Mixup	2.9268%	-2.7796%	-0.4878%	-1.9101%	-1.9512%	-24.1470%
	EMixup (our)	4.3902%	1.4576%	1.9512%	-0.9324%	-7.3171%	-6.9484%
SelfRegulationSCP1	Baseline	88.0546%	88.0407%	91.4676%	91.4576%	92.8328%	92.8244%
	Jitter	0.0000%	0.0050%	-3.7543%	-3.7559%	0.0000%	-0.0037%
	Hflip	0.3413%	0.3443%	-4.0956%	-4.1003%	0.0000%	0.0030%
	Vflip	-5.8020%	-5.7983%	-12.9693%	-13.0236%	-3.0717%	-3.0835%
	Scaling	-1.0239%	-1.0440%	-3.7543%	-3.7617%	0.0000%	0.0000%
	Window_warp	**5.1195%**	**5.1269%**	-5.8020%	-5.9996%	0.0000%	-0.0080%
	Window_slice	3.7543%	3.7605%	-1.7065%	-1.7110%	0.0000%	-0.0037%
	Permutation	4.4369%	4.4500%	-3.4130%	-3.4080%	-0.6826%	-0.6977%
	Mixup	0.6826%	0.6918%	-4.4369%	-4.4452%	-5.8020%	-5.7975%
	EMixup (our)	-0.3413%	-0.3390%	-1.3652%	-1.3557%	-3.0717%	-3.0634%
SelfRegulationSCP2	Baseline	52.7778%	52.6008%	51.1111%	51.0143%	52.2222%	52.1986%
	Jitter	0.0000%	-0.1532%	-2.2222%	-2.2837%	0.0000%	0.0000%
	Hflip	1.6667%	1.8437%	1.6667%	1.4332%	**4.4444%**	**4.0303%**
	Vflip	**4.4444%**	**4.0312%**	-1.1111%	-2.6462%	3.3333%	3.2690%
	Scaling	-0.5556%	-0.3844%	-1.1111%	-1.0205%	-1.1111%	-1.2389%
	Window_warp	1.1111%	1.2525%	**2.7778%**	**2.6328%**	2.7778%	2.6886%
	Window_slice	1.1111%	1.2753%	0.5556%	0.3989%	3.3333%	3.2690%
	Permutation	-1.6667%	-2.1008%	-1.6667%	-2.5643%	2.7778%	2.8000%
	Mixup	0.5556%	0.7326%	0.5556%	0.6508%	-2.2222%	-18.8653%
	EMixup (our)	0.5556%	0.7268%	**2.7778%**	1.0626%	-1.6667%	-17.6425%

5. Conclusion

This study has systematically explored the effectiveness of various data augmentation techniques in time series analysis, focusing on forecasting and classification tasks. We designed comprehensive experiments to evaluate augmentation techniques from two primary perspectives: their ability to enhance performance across various models, data and their effectiveness in outperforming other methods. The proposed EMixup offers several practical benefits for real-world time series analysis, demonstrating broad

applicability across various models, diverse datasets (power management, ...), and tasks. Its relatively simple implementation makes it accessible to practitioners, and its ability to generate synthetic data and enhances model robustness by incorporating a Generating Noise Module. While EMixup appears scalable to high-dimensional time series due to its noise generation process and does not impact inference time for real-time applications, it exhibits potential limitations such as sensitivity to hyperparameter settings, including the skewed Beta distribution, and a possible reduction in the diversity of synthetic samples due to the emphasis on preserving original time series structure. Furthermore, the risk of introducing unrealistic patterns and the need for dataset-specific tuning, should be considered for effective deployment. Further research is needed to fully assess its scalability to large datasets and to develop learnable hyperparameters (adaptive Beta distribution).

Declarations

Author contribution. The first author has a significant contribution on this research, and other authors have contributed equally for different parts of this paper. All authors read and approved the final paper.
Funding statement. This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.
Conflict of interest. The authors declare no conflict of interest.
Additional information. No additional information is available for this paper.

Data and Software Availability Statements

The source code is available at https://github.com/khoanta-ai/TS_EMixup.

References

[1] A. Zeng, M. Chen, L. Zhang, and Q. Xu, "Are Transformers Effective for Time Series Forecasting?," *Proc. AAAI Conf. Artif. Intell.*, vol. 37, no. 9, pp. 11121–11128, Jun. 2023, doi: 10.1609/aaai.v37i9.26317.

[2] H. Wu, T. Hu, Y. Liu, H. Zhou, J. Wang, and M. Long, "TimesNet: Temporal 2D-Variation Modeling for General Time Series Analysis," *11th Int. Conf. Learn. Represent. ICLR 2023*, pp. 1–23, Oct. 2022. [Online]. Available at: https://arxiv.org/pdf/2210.02186.

[3] D. Dera, S. Ahmed, N. C. Bouaynaya, and G. Rasool, "TRustworthy Uncertainty Propagation for Sequential Time-Series Analysis in RNNs," *IEEE Trans. Knowl. Data Eng.*, vol. 36, no. 2, pp. 1–13, Feb. 2023, doi: 10.1109/TKDE.2023.3288628.

[4] Y. Liu *et al.*, "iTransformer: Inverted Transformers Are Effective for Time Series Forecasting," *12th Int. Conf. Learn. Represent. ICLR 2024*, pp. 1–25, Oct. 2023. [Online]. Available at: https://arxiv.org/pdf/2310.06625.

[5] Z. Wang *et al.*, "A Comprehensive Survey on Data Augmentation," *arXiv*, pp. 1–20, May 2024. [Online]. Available at: https://arxiv.org/pdf/2405.09591.

[6] A. Lazcano, M. A. Jaramillo-Morán, and J. E. Sandubete, "Back to Basics: The Power of the Multilayer Perceptron in Financial Time Series Forecasting," *Mathematics*, vol. 12, no. 12, p. 1920, Jun. 2024, doi: 10.3390/math12121920.

[7] R. Espinosa, F. Jiménez, and J. Palma, "Embedded feature selection in LSTM networks with multi-objective evolutionary ensemble learning for time series forecasting," *arXiv*, pp. 1–25, Dec. 2023. [Online]. Available at: https://arxiv.org/pdf/2312.17517.

[8] W. Chen and J. Li, "SIGAN-CNN: Convolutional Neural Network Based Stepwise Improving Generative Adversarial Network for Time Series Classification of Small Sample Size," *IEEE Access*, vol. 12, pp. 85499–85510, 2024, doi: 10.1109/ACCESS.2024.3413948.

[9] A. Das, W. Kong, R. Sen, and Y. Zhou, "A decoder-only foundation model for time-series forecasting," *Proc. Mach. Learn. Res.*, vol. 235, pp. 10148–10167, Oct. 2023. [Online]. Available at: https://arxiv.org/pdf/2310.10688.

[10] V. Naghashi, M. Boukadoum, and A. B. Diallo, "A multiscale model for multivariate time series forecasting," *Sci. Rep.*, vol. 15, no. 1, p. 1565, Dec. 2025, doi: 10.1038/S41598-024-82417-4.

[11] B. Liu, Z. Zhang, and R. Cui, "Efficient Time Series Augmentation Methods," in *2020 13th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)*, Oct. 2020, pp. 1004–1009, doi: 10.1109/CISP-BMEI51763.2020.9263602.

[12] W. Yang, J. Yuan, and X. Wang, "SFCC: Data Augmentation with Stratified Fourier Coefficients Combination for Time Series Classification," *Neural Process. Lett.*, vol. 55, no. 2, pp. 1833–1846, Apr. 2023, doi: 10.1007/s11063-022-10965-9.

[13] M. Goubeaud, N. Gmyrek, F. Ghorban, L. Schelkes, and A. Kummert, "Random Noise Boxes: Data Augmentation for Spectrograms," in *2021 IEEE International Conference on Progress in Informatics and Computing (PIC)*, Dec. 2021, pp. 24–28, doi: 10.1109/PIC53636.2021.9687058.

[14] W. Zhao and L. Fan, "Time-series representation learning via Time-Frequency Fusion Contrasting," *Front. Artif. Intell.*, vol. 7, p. 1414352, Jun. 2024, doi: 10.3389/frai.2024.1414352.

[15] Y. Deng, R. Liang, D. Wang, A. Li, and F. Xiao, "Decomposition-based Data Augmentation for Time-series Building Load Data," in *Proceedings of the 10th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*, Nov. 2023, pp. 51–60, doi: 10.1145/3600100.3623727.

[16] B. Kalina, J.-H. Lee, and K.-T. Na, "Enhancing Portfolio Performance through Financial Time-Series Decomposition-Based Variational Encoder-Decoder Data Augmentation," *Symmetry (Basel).*, vol. 16, no. 3, p. 283, Feb. 2024, doi: 10.3390/sym16030283.

[17] X. Wu, D. Zhang, G. Li, X. Gao, B. Metcalfe, and L. Chen, "Data augmentation for invasive brain–computer interfaces based on stereo-electroencephalography (SEEG)," *J. Neural Eng.*, vol. 21, no. 1, p. 016026, Feb. 2024, doi: 10.1088/1741-2552/ad200e.

[18] L. Wang, L. Zeng, and J. Li, "AEC-GAN: Adversarial Error Correction GANs for Auto-Regressive Long Time-Series Generation," *Proc. AAAI Conf. Artif. Intell.*, vol. 37, no. 8, pp. 10140–10148, Jun. 2023, doi: 10.1609/aaai.v37i8.26208.

[19] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "Mixup: Beyond Empirical Risk Minimization," *6th Int. Conf. Learn. Represent. ICLR 2018 - Conf. Track Proc.*, pp. 1–13, Oct. 2017, 2025. [Online]. Available at: https://arxiv.org/pdf/1710.09412.

[20] A. N. E. Xplorer and A. F. Ramework, "iGraphMix: Input Graph Mixup Method For Node Classification," *arXiv*, pp. 1–11, 2023, [Online]. Available at: https://openreview.net/forum?id=a2ljjXeDcE&noteId=a2ljjXeDcE.

[21] Z. Liu, S. Li, G. Wang, C. Tan, L. Wu, and S. Z. Li, "Harnessing Hard Mixed Samples with Decoupled Regularizer," in *Advances in Neural Information Processing Systems*, 2022, no. DM, pp. 1–23, [Online]. Available at: http://arxiv.org/abs/2203.10761.

[22] M. Kang, M. Kang, and S. Kim, "Catch-Up Mix: Catch-Up Class for Struggling Filters in CNN," *Proc. AAAI Conf. Artif. Intell.*, vol. 38, no. 3, pp. 2705–2713, Mar. 2024, doi: 10.1609/aaai.v38i3.28049.

[23] M. Kang and S. Kim, "GuidedMixup: An Efficient Mixup Strategy Guided by Saliency Maps," *Proc. AAAI Conf. Artif. Intell.*, vol. 37, no. 1, pp. 1096–1104, Jun. 2023, doi: 10.1609/aaai.v37i1.25191.

[24] S. Hong, Y. Yoon, H. Joo, and J. Lee, "GBMix: Enhancing Fairness by Group-Balanced Mixup," *IEEE Access*, vol. 12, pp. 18877–18887, 2024, doi: 10.1109/ACCESS.2024.3358275.

[25] J. Maurya, K. R. Ranipa, O. Yamaguchi, T. Shibata, and D. Kobayashi, "Domain Adaptation using Self-Training with Mixup for One-Stage Object Detection," in *2023 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, Jan. 2023, pp. 4178–4187, doi: 10.1109/WACV56688.2023.00417.

[26] D. Ryou, I. Ha, H. Yoo, D. Kim, and B. Han, "Robust Image Denoising Through Adversarial Frequency Mixup," in *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2024, pp. 2723–2732, doi: 10.1109/CVPR52733.2024.00263.

[27] K. Aggarwal and J. Srivastava, "Embarrassingly Simple Mixup for Time-series," *arXiv*, pp. 1–11, Apr. 2023. [Online]. Available at: https://arxiv.org/pdf/2304.04271.

[28] P. Guo, H. Yang, and A. Sano, "Empirical Study of Mix-based Data Augmentation Methods in Physiological Time Series Data," in *2023 IEEE 11th International Conference on Healthcare Informatics (ICHI)*, Jun. 2023, pp. 206–213, doi: 10.1109/ICHI57859.2023.00037.

[29] L. N. Darlow, A. Joosen, M. Asenov, Q. Deng, J. Wang, and A. Barker, "TSMix: time series data augmentation by mixing sources," in *Proceedings of the 3rd Workshop on Machine Learning and Systems*, May 2023, pp. 109–114, doi: 10.1145/3578356.3592584.

[30] H. Zhou *et al.*, "Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting," *Proc. AAAI Conf. Artif. Intell.*, vol. 35, no. 12, pp. 11106–11115, May 2021, doi: 10.1609/aaai.v35i12.17325.

[31] A. Bagnall *et al.*, "The UEA multivariate time series classification archive, 2018," *arXiv*, pp. 1–36, Oct. 2018. [Online]. Available at: https://arxiv.org/pdf/1811.00075.