ELITS - Efficient Lightweight Imputation for Time Series

Pranav Sastry[1], **Kalyan Reddy**[1], **Sumanta Mukherjee**[2],
Vijay Ekambaram[2], **Pankaj Dayama**[2], **Prathosh AP**[1]

[1]Indian Institute of Science, Bengaluru
[2]IBM Research

Abstract

Time-series imputation is a crucial preprocessing step for refining raw data to ensure accurate analysis, as missing values in multivariate time-series can obscure patterns and lead to erroneous predictions. The challenge lies in efficiently learning the complex correlations between variates over time to fill these missing values. Existing solutions often fall short by either focusing solely on univariate data or by significantly inflating the model size as the number of variables increases in multivariate imputation modeling. To address these limitations, we introduce ELITS, a novel lightweight MLP-Mixer based architecture which effectively handles multivariate correlations while maintaining a compact model size, regardless of the number of variates involved. ELITS achieves **20.97%** performance improvement over the current state of the art while using **858x** fewer parameters. Our comprehensive evaluations showcase ELITS's robustness against various missing value patterns and demonstrate its practical utility when used to handle missing values in forecasting, representing a significant advancement over existing techniques.

Introduction

Time series analysis, particularly in the context of multivariate time series, plays a pivotal role in various enterprise settings, including manufacturing, retail, energy and utilities. For instance, real-time analysis of industrial processes by leveraging edge computing to predict future states based on sensor data can help support applications such as predictive maintenance and process optimization. To ensure efficiency, prediction models must be lightweight, thus minimizing computational overhead while maintaining high performance.

However, typical time series data in enterprise applications often have many missing entries due to various reasons, including data collection issues and communication breakdowns in the network. These missing values pose significant challenges, as they can degrade the performance of downstream tasks including forecasting, classification, and anomaly detection. Thus, efficient imputation techniques are essential to handle missing values in streaming data effectively, ensuring robust and reliable real-time downstream predictions. However, it's a complex challenge due to the temporal dependencies inherent in time series data. In multivariate time series, the problem intensifies as missing values in one variable (also referred to as channel) can impact others due to inter-relationships. Capturing these complex channel correlations is essential for effective imputation, requiring advanced techniques that model both temporal and cross-channel dependencies. Failing to address missing values adequately can lead to erroneous insights and suboptimal predictions.

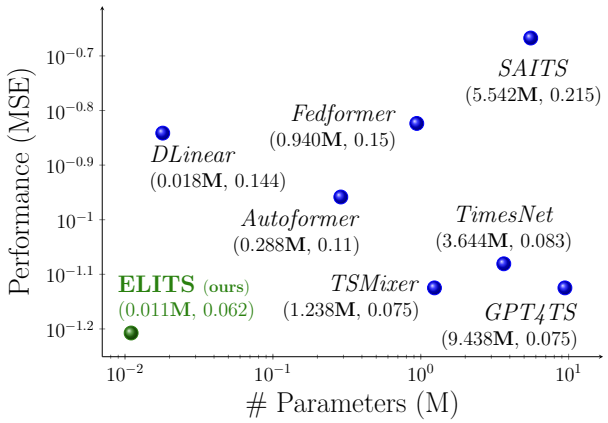


Figure 1: Comparison of different time series imputation models in terms of Mean Squared Error (MSE) and the Number of Parameters (in millions). Lower MSE values indicate better performance. ELITS, achieves the lowest MSE while maintaining the smallest number of parameters, demonstrating its efficiency compared to other state of the art models such as GPT4TS, TimesNet, and SAITS.

Existing deep learning approaches face critical scalability challenges: attention-based models incur quadratic complexity in both sequence length and number of channels, while MLP mixers suffer quadratic parameter growth with the number of channels. These limitations become prohibitive for real-world datasets with hundreds of channels (e.g., Traffic: 862, Electricity: 321), necessitating lightweight architectures with sub-quadratic scaling.

Our Contributions

We propose ELITS, a novel MLP-Mixer architecture that enhances multivariate time series imputation through three key design innovations:

- **Channel Patching:** Existing MLP-Mixer architectures suffer from quadratic parameter growth as channels increase, rendering them infeasible for high-dimensional data. We introduce channel patching, creating non-overlapping patches along both time and channel dimensions. This approach reduces model complexity while maintaining performance. As shown in Figure 6, our design ensures constant parameter count regardless of channel count—ELITS maintains a flat scaling curve while baseline methods like TSMixer exhibit exponential growth—a critical advantage for multivariate modeling.
- **Mixers In Parallel:** Unlike conventional sequential mixer operations across channels, patches, and features, we apply these operations in parallel. This design enhances both modeling accuracy and compactness for imputation tasks, contributing to our $858\times$ parameter reduction compared to state-of-the-art methods.
- **Head Mixer:** We introduce a head mixer block that learns correlations across parallel mixer outputs, enabling effective aggregation while maintaining reduced model size. This innovative paradigm of parallel operations followed by head mixing significantly improves multivariate imputation accuracy.

Through extensive evaluation, ELITS achieves 20.97% performance improvement over state-of-the-art methods while demonstrating robustness across various missing value patterns and practical utility in downstream forecasting tasks. As shown in Figure 1, ELITS achieves the lowest MSE with minimal computational overhead.

Related Work

To address the missing value imputation problem several popular approaches have been developed. Traditional methods such as mean imputation and linear interpolation offer simplicity but often fail to capture complex temporal and cross-channel dependencies. Various statistical and machine-learning methods have been proposed for modeling such dependencies. Recent advancements in missing value imputation have used more advanced techniques and architectures, including CNNs (Wu et al. 2023; Khan, Wang, and Liu 2022; Wang and Oates 2015), RNNs (Cao et al. 2018), Transformer-based architectures (Wu et al. 2021; Kitaev, Kaiser, and Levskaya 2020; Zhou et al. 2021, 2022; Du, Côté, and Liu 2023; Zhou et al. 2023; Woo et al. 2022; Liu et al. 2022), and deep generative models (Qin and Wang 2023; Alcaraz and Strodthoff 2022; Luo et al. 2018; Miao et al. 2021). TimesNet (Wu et al. 2023) explored the use of convolutional neural networks (CNNs) for time series imputation by transforming the data into a 2D tensor representation. GPT4TS (Zhou et al. 2023) show that NLP pretrained transformer model performs well on various downstream time series analysis tasks including imputation task with fine-tuning. In recent years, several comprehensive surveys have been conducted on time series imputation, highlighting various trends and advancements in the field (Wang et al. 2024a; Fang and Wang 2020; Adhikari et al. 2022).

Existing solutions often fall short in two critical areas. Firstly, many methods focus exclusively on univariate data, addressing missing values within a single channel without considering the interactions between multiple channels over time. This univariate approach limits the ability to capture complex cross channel dependencies that are crucial for accurate imputation in multivariate time series data. Existing lightweight architectures proposed for forecasting (Lee, Park, and Lee 2024; Lin et al. 2024) follows this univariate approach. Secondly, while some methods do attempt to handle multivariate imputation, they often do so at the cost of significantly inflating the model size. As the number of channels increases, these models become increasingly complex and resource-intensive, making them impractical for real-world applications involving streaming data where computational efficiency and scalability are crucial. This trade-off between model complexity and the ability to accurately impute missing values across multiple channels remains a significant challenge in the field. To address these limitations, we introduce ELITS, a novel lightweight MLP-Mixer (Tolstikhin et al. 2021) based architecture that effectively handles cross-channel correlations while maintaining a compact model size, independent of the number of channels involved.

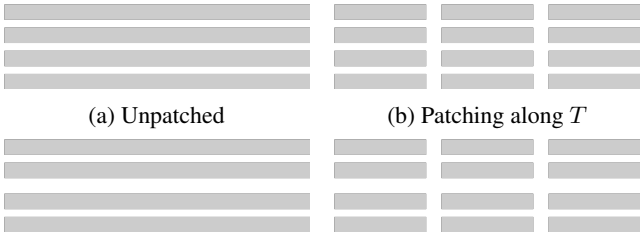
MLP-Mixer based architectures have recently garnered significant attention in the time-series community due to their ability to balance accuracy and computational efficiency. Notable models, such as TSMixer (Ekambaram et al. 2023), TimeMixer (Wang et al. 2024b), and TTM (Ekambaram et al. 2024), have demonstrated the effectiveness of these architectures compared to traditional transformer models on the forecasting task. The core advantage of MLP-Mixer models lies in their simplicity and efficiency. Additionally, several enhancements to the basic MLP-Mixer architecture, as proposed in recent literature, have further improved their performance. For example, TSMixer introduces gating attention and other reconciliation techniques, while TimeMixer employs decomposition methods prior to mixing. These innovations enable the models to maintain a compact size while enhancing accuracy, making them strong candidates for time-series imputation tasks.

ELITS Methodology

Multivariate Time Series Imputation involves filling in missing values in a given time series data comprising of multiple channels. We consider observation sequences of fixed length T. Hence, every data sample (X) can be viewed as a $C \times T$ matrix, where C is the number of channels and T represents the sequence length. Values are said to be missing if no value is recorded at the t^{th} time step in the c^{th} channel.

Training Workflow

During training, we simulate missing values in the input X using a randomly generated boolean mask ($M \in$



(a) Unpatched (b) Patching along T

(c) Patching along C (d) Patching along C and T

Figure 2: Different patching strategies applied to multivariate time series. Unpatched inputs (C, T) are first patched along T to yield N_T number of sequence patches, each containing L_T time steps resulting in sequence patched inputs (C, N_T, L_T). These are then patched along C to yield N_C number of channel patches, each containing L_C number of channels, resulting in sequence and channel patched inputs (N_C, L_C, N_T, L_T)

$\{0, 1\}^{C \times T}$), where 1 indicates a value is missing and 0 otherwise. We consider the values to be Missing Completely At Random (MCAR) (Wang et al. 2024a), where missing values are independent of any other values. $\tilde{X} = X \odot (1 - M)$ is a simulated MCAR instance of X, where 0 represents missing values, is fed into an imputation model.

ELITS, a deep learning based imputation model, parameterized by $\mathbf{\Theta}$, produces $\hat{X} = \mathbf{\Theta}(\tilde{X})$, such that $\hat{X} \approx X$. We train the model $\mathbf{\Theta}$ with the masked modeling approach whose objective is to minimize the mean squared reconstruction error at the masked indices (Eq. 1). Matrix entries at the c^{th} channel and the t^{th} time step are denoted as $(\cdot)_{ct}$.

$$\boldsymbol{\mathcal{E}}(X, \hat{X}) = \frac{\sum_{c \in C} \sum_{t \in T} \mathbb{1}[M_{ct} = 1] \cdot (X_{ct} - \hat{X}_{ct})^2}{\sum_{c \in C} \sum_{t \in T} M_{ct}} \quad (1)$$

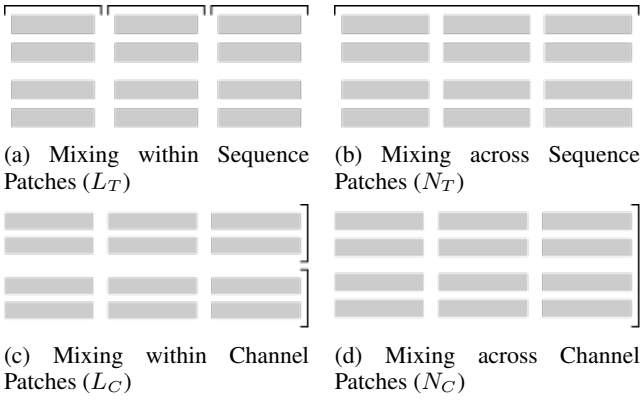
ELITS Architecture

We propose a novel, lightweight MLP based mixer architecture for effcient time series imputation (Figure 4).

Instance Normalization Masked time series data $\tilde{X}$, undergoes a learnable reversible instance normalization (RevIN) (Kim et al. 2022), which standardizes the data by adjusting for mean and scaling by standard deviation. This normalization is specifically applied only to the unmasked time steps to effectively handle data shifts within the time series.

Sequence Patching We segment the normalized input $\tilde{X} \in \{C, T\}$ (Figure 2a) along the sequence length T into N_T number of non-overlapping patches, each containing L_T time steps, yielding $\tilde{X}_p \in \{C, N_T, L_T\}$ (Figure 2b). This patching technique, introduced by Nie et al. (2023), is widely adopted in deep learning based time series models.

Channel Patching Existing mixer models that capture cross channel correlations suffer from a quadratic increase in the number of parameters as the number of channels increases, which is exacerbated in datasets where the number channels is in the order of hundreds. To address this, we introduce Channel Patching, involving patching along the channels C. While sequence patching is common, channel patching hasn't been explored, most likely due to lack of semantic meaning in channel ordering. However, our findings indicate that mixer operations applied to channel patches is robust to channel ordering (Subsection Ablation), effectively capturing local and global information. Channel Patching, when realized results in N_C number of non-overlapping channel patches, each containing L_C number of channels (Figure 2c). We utilize both sequence and channel patching by applying channel patching on top of the already sequence patched inputs, yielding $\tilde{X}_p \in \{N_C, L_C, N_T, L_T\}$ (Figure 2d).



(a) Mixing within Sequence Patches (L_T) (b) Mixing across Sequence Patches (N_T)

(c) Mixing within Channel Patches (L_C) (d) Mixing across Channel Patches (N_C)

Figure 3: Four types of patch mixing operations. (a) Intra-sequence patch mixing: Mixes patches within the same channel across time. (b) Inter-sequence patch mixing: Mixes patches across different time steps. (c) Intra-channel patch mixing: Mixes patches within the same channel group. (d) Inter-channel patch mixing: Mixes patches across different channel groups.

Mixing Mixing is the process of learning correlations between features along a particular data dimension, typically realized through a simple MLP, which consists of one or more linear layers followed by non-linear activation functions. Given patched inputs $\tilde{X}_p \in \{N_C, L_C, N_T, L_T\}$, in order to perform effective imputation, it is necessary for mixing to be applied along all dimensions, which aids the model in capturing correlations between features within a patch (Intra Patch) and correlations between features from across patches (Inter Patch). The layers that perform the mixing operation are known as Mixers. Four such mixers can be realized on sequence and channel patched inputs — Intra Sequence Patch Mixer (**Intra SPM**) operates on L_T (Figure 3a), Inter Sequence Patch Mixer (**Inter SPM**) operates on N_T (Figure 3b), Intra Channel Patch Mixer (**Intra CPM**) operates on L_C (Figure 3c) and finally, Inter Channel Patch Mixer (**Inter CPM**) operates on N_C (Figure 3d). For the sake of generality, when it is not necessary to distinguish between the sequence and channel patches, we use the terms Intra Patch Mixer (Intra PM) and Inter Patch Mixer (Inter PM). Intra PMs that aid in learning lower level features at the time step level, is more effectively learnt in dimensions

that are higher than the data dimension, hence while mixing within patches, we make use of a projection to a latent dimension D which is typically 2-3$\times$ the data dimension and once mixing is complete, we project it back to the data space without preserving D (Subsection Ablation). Shape of the inputs to, and, outputs from, each of the mixers is maintained to be the same shape as $\tilde{X}_p$ (N_C, L_C, N_T, L_T).

ELITS Layer Each layer in ELITS comprises of Intra SPM (Figure 4a), Inter SPM (Figure 4b), Intra CPM (Figure 4c) and Inter CPM (Figure 4d) organized novelly in a multi-headed fashion with each of the mixers as the heads, all of them operating in parallel. Each mixer is isolated from the influence of other mixers unlike existing mixer models that stack one mixer on top of the other. We hypothesised that such stacking of mixers could lead to one or more of them negatively influencing the others and validate this empirically through ablation (Subsection Ablation). Mixers when stacked results in the model implicitly learning combined information from individual mixers which is vital to capturing the underlying dynamics of complex patterns, but there is no such notion when mixers are isolated naively. We introduce a novel **Head Mixer** (Figure 4e) to allow the model to explicitly learn combined information from the isolated heads. Head Mixer operates by stacking outputs from each of the heads, followed by N_Y MLPs that helps mix information from the individual heads. Residual connections are introduced to promote gradient flow and to ensure robustness across noisy head correlations.

ELITS N_X number of ELITS Layers are stacked to allow the model to learn more complex features as the depth increases. The output from the final ELITS layer is fed into **Head Accumulator** (Figure 4f) that pools outputs from all of the heads to yield a single output which captures all of the essential correlations required in order to effectively impute. Head Accumulator operates by stacking outputs from each of the heads, followed by an MLP with a couple of layers to bring it back to the data space. Inverse Affine RevIN denormalizes the outputs to yield the final reconstructed inputs with the missing values filled. For more implementation details and pseudocode, kindly refer to the supplementary section.

Experiments

Experimental Setup

Datasets We assess the performance of ELITS using the following seven well known multivariate datasets — ETTh1, ETTh2, ETTm1, ETTm2, Weather, Electricity and Traffic. These datasets have been widely adopted in the literature (Zhou et al. 2021) (Zhou et al. 2022) (Ekambaram et al. 2023) (Nie et al. 2023) for benchmarking multivariate time-series models and can be publicly accessed via Wu et al. (2023); thuml (2023). For consistency, we adhere to the same data loading parameters, including the train/validation/test split ratios, as outlined in Wu et al. (2023).

Baselines For benchmarking and evaluation, we consider the following — Foundation Models[1] (GPT4TS (Zhou et al. 2023) (**SOTA**), TimesNet (Wu et al. 2023)), MLP Mixers (TSMixer (Ekambaram et al. 2023)), Imputation Specific Models (SAITS (Du, Côté, and Liu 2023)) , Transformers (Autoformer (Wu et al. 2021), FedFormer (Zhou et al. 2022)), Single Layer Models (DLinear (Zeng et al. 2022)) and Classical Methods (Mean (Mean Fill), Linear (Linear Interpolation)).

Experiment Configuration We train ELITS on four different missing value ratios (0.125, 0.25, 0.375, and 0.5) and report Mean Squared Error (MSE) and/or Mean Absolute Error (MAE) values on the test set at the masked indices. All experiments are run with 3 different random seeds and the average MSE and/or MAE values across seeded runs are reported.

Training and Hardware Configuration All models are optimized with Adam (Kingma and Ba 2014) for 25 epochs, with batch-size as 16 and an initial learning rate of 0.01 as scheduled by Cosine Annealing LR scheduler (Loshchilov and Hutter 2017). Additional details on hyperparameters can be found in the supplementary. All experiments are carried out on a single NVIDIA A6000 GPU with 48GB RAM.

Mask Generation and Padding Mask (M) is dynamically generated for each example in train/validation/test sets. Recent research in time series imputation (Zhou et al. 2023; Wu et al. 2023) has been focused towards values that are missing at random time steps across channels and we refer to the mask that reflects this missingness as Point Masking (Figure 5a). Unless specified, M conforms to point masking which is also what we focus on. While patching along channels, it is required that the number of channels C be divisible by channel patch length L_C. However, this might not always be satisfied, hence, additional channels are added via zero padding. To ensure a fair comparison with other baselines, M does not include time steps from the padded channels as predicting zeros becomes trivial for the model.

Imputation Benchmarks

Point Masking Results Table 1 compares the performance of ELITS with other baselines on point masking. On an average, across all datasets and missing value ratios, ELITS (with only 11K parameters) outperforms the existing state of the art by 20.97% in terms of MSE.

Efficiency Metrics The efficiency of ELITS is demonstrated with a focus on parameter and computational efficiency. Number of trainable parameters (Params), floating point operations per iteration (FLOPS) and multiple accumulate operations (MACS) are reported. As shown in Table 2, ELITS significantly outperforms GPT4TS by 20.97% in terms of performance despite having $858\times$ lesser number

[1]We only consider foundation models that train imputation from scratch and is computationally feasible to be trained. UniTS (Gao et al. 2024) performs few-shot imputation whereas Moment (Goswami et al. 2024) was computationally expensive to reproduce.

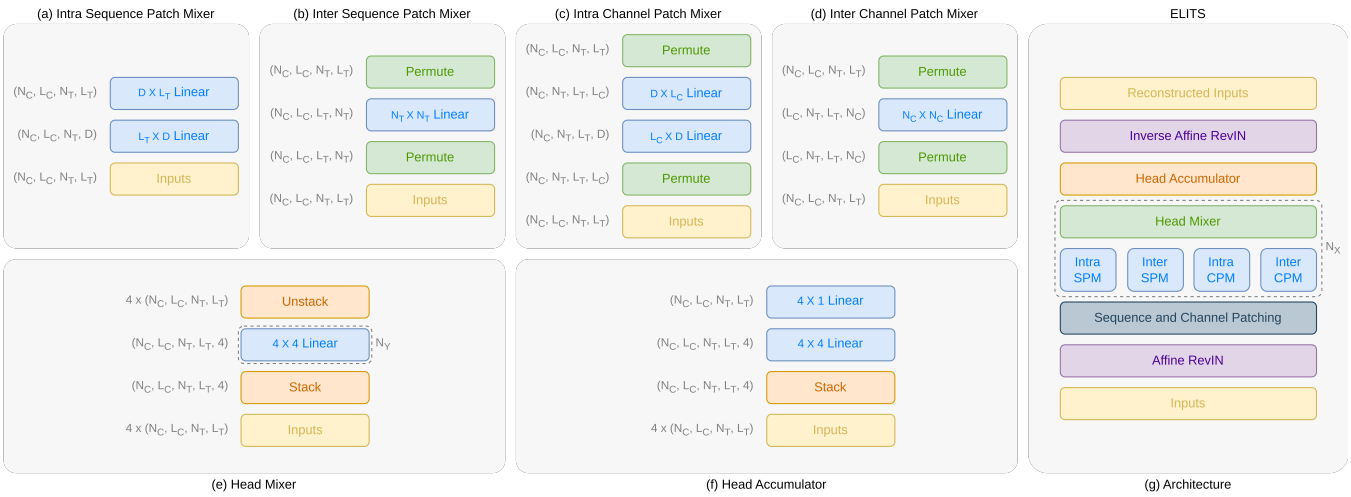


Figure 4: Architecture of ELITS. Each layer of ELITS consists of Intra and Inter Patch Mixers (heads) applied along sequence and channel patches in parallel. Head Mixer combines information from different types of mixers. ELITS consists of N_X such layers with residual connections in between to aid gradient flow. Head Accumulator accumulates information from all heads to produce the reconstructed inputs. Each Linear layer consists of a single fully connected layer followed by a GELU activation function. Learnable Affine RevIN helps tackle distribution shift.

Table 1: Results on point masking. MSE and MAE are reported on all models averaged over four different missing value ratios (0.125, 0.25, 0.375 and 0.5) % Imp denotes the percentage improvement in performance of ELITS over other baselines. Indicators - Best: **Teal**, Second Best: **Black**

Dataset	ELITS		GPT4TS		TimesNet		TSMixer		SAITS		Autoformer		Fedformer		DLinear		Mean		Linear	
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETTh1	**0.049**	**0.144**	0.069	0.173	0.078	0.187	0.066	0.172	**0.063**	**0.168**	0.103	0.214	0.116	0.246	0.201	0.306	0.714	0.563	0.599	0.491
ETTh2	**0.043**	**0.128**	**0.048**	**0.141**	0.05	0.146	0.055	0.151	0.319	0.407	0.056	0.156	0.163	0.279	0.142	0.260	0.355	0.388	0.494	0.440
ETTm1	**0.024**	**0.098**	**0.027**	**0.105**	0.028	0.109	0.028	0.107	0.031	0.126	0.054	0.156	0.066	0.185	0.099	0.213	0.706	0.552	0.420	0.387
ETTm2	**0.021**	**0.081**	**0.021**	**0.085**	**0.022**	0.089	0.024	0.091	0.339	0.428	0.029	0.105	0.101	0.215	0.096	0.208	0.231	0.308	0.297	0.311
weather	**0.027**	**0.041**	0.031	0.057	**0.03**	**0.054**	0.034	0.072	0.039	0.063	0.031	0.057	0.099	0.203	0.053	0.110	0.216	0.271	0.338	0.335
electricity	**0.051**	**0.145**	0.091	0.207	0.092	0.211	**0.061**	**0.165**	0.2	0.303	0.101	0.225	0.130	0.260	0.132	0.261	0.864	0.769	0.414	0.417
traffic	**0.217**	**0.191**	**0.239**	0.227	0.282	0.248	0.257	**0.19**	0.512	0.252	0.401	0.332	0.375	0.313	0.285	0.292	1.432	0.813	1.293	0.589
Best Count	7	6	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
% Imp	-	-	20.97	20.34	33.87	26.27	20.97	14.41	246.77	111.02	77.42	50.85	141.94	105.93	132.26	99.15	940.32	344.07	788.71	259.32

of parameters. We also outperform all the other models (except DLinear) in terms of computational efficiency metrics by a noticeable margin. Although DLinear excels in computational efficiency metrics due to its simplistic architecture, its performance is notably inferior, dropping by 132% compared to ELITS despite having $1.7\times$ more number of parameters than ELITS.

Robustness Across Different Masking Types We evaluate ELITS across four masking patterns that represent different real-world scenarios: **Point masking** (scattered individual missing values), **Block masking** (contiguous temporal gaps simulating sensor failures), **Blackout masking** (complete channel loss requiring full reliance on cross-channel correlations), and **Combined masking** (realistic mixed patterns). Table 4 shows ELITS maintains robust performance across all patterns, with particularly strong results on blackout masking due to our channel patching strategy. Block length is fixed at 4 time steps.

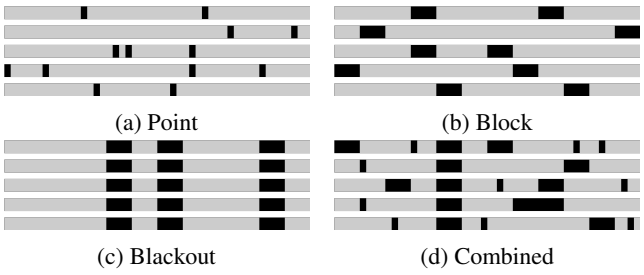


Figure 5: Different Masking Types. The horizontal dimension indicates time steps and the vertical dimension indicates channels. Masked time steps are highlighted in black.

Table 2: Results indicating parameter (**Params**) and computational (**FLOPS**, **MACS**) efficiencies of different models along with their respective MSE (**MSE**) on point masking across datasets are reported. Units of different efficiency metrics are mentioned beneath their respective metrics. Indicators - Best: **Teal**, Second Best: **Black**, Increase in metric value compared to ELITS: **Purple**

Model	Params (K)		FLOPS (G/iter)	MACS (G/iter)	MSE	
ELITS	**11**	-	**0.9**	**0.4**	**0.062**	-
GPT4TS	9438	**858×**	58.8	30.0	**0.075**	**20.97%**
TimesNet	3645	**331×**	47.8	24.0	0.083	**33.87%**
TSMixer	1239	**113×**	11.9	6.0	**0.075**	**20.97%**
SAITS	5542	**504×**	18.2	9.1	0.215	**246.77%**
Autoformer	289	**26×**	**0.9**	**0.4**	0.11	**77.42%**
Fedformer	940	**85×**	**0.9**	**0.4**	0.15	**141.94%**
DLinear	**19**	**1.7×**	**0.2**	**0.08**	0.144	**132.26%**

Table 3: Results indicating the tradeoff between number of parameters (**Params**) and MSE (**MSE**), with and without channel patching (CP)

Dataset	with CP		without CP	
	Params	**MSE**	**Params**	**MSE**
weather	10345	0.027	15745	0.028
electricity	20513	0.051	174323	0.049
traffic	23809	0.217	342981	0.212

Table 4: Results demonstrating robustness of ELITS in handling different masking types. Only top performing baselines are reported for compactness. Average MSE and MAE across missing value ratios and datasets are reported on each of the four mask types (point, block, blackout and combined) along with average across mask types (AVG). Indicators - Best: **Teal**, Second Best: **Black**.

Type	ELITS		GPT4TS		TimesNet		TSMixer	
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
point	**0.062**	**0.118**	**0.075**	0.142	0.083	0.149	**0.075**	**0.135**
block	**0.106**	**0.16**	0.339	0.200	0.119	0.189	**0.114**	**0.165**
blackout	**0.138**	**0.19**	0.379	0.248	**0.139**	**0.203**	0.15	**0.203**
combined	**0.101**	**0.158**	0.3	0.198	**0.113**	0.182	0.118	**0.176**
AVG	**0.102**	**0.157**	0.273	0.197	**0.114**	0.181	**0.114**	**0.17**

Forecasting With Missing Values

Time series forecasting with missing input values typically uses filling strategies: Zero Fill, Forward/Backward Fill, Mean/Median Fill, or Impute Fill using dedicated imputation models. While Impute Fill generally yields the most accurate forecasts, existing imputation models have hundreds of thousands to millions of parameters, making them computationally expensive for online tasks.

We evaluate ELITS as an Impute Fill model by training PatchTST (Nie et al. 2023) forecasting models with different filling techniques across four missing value ratios (0.125, 0.25, 0.375, 0.5). Table 5 shows ELITS consistently outperforms other filling methods, with notable 7-8% performance gains on high-dimensional datasets like electricity (321 channels) and traffic (862 channels), demonstrating that traditional filling techniques are inadequate for such datasets.

Table 5: Results comparing the effectiveness of different fill types — Zero Fill (**ZF**), Forward Fill (**FF**), Backward Fill (**BF**), Mean Fill (**MeanF**), Median Fill (**MedF**) and Impute Fill (**ELITS**) in handling missing values on the forecasting task. Average MSE across missing value ratios with point masking, along with forecasting performance without any missing values (**w/o MV**) are reported. % Imp denotes percentage improvement in performance of ELITS over other fill types. Indicators - Best: **Teal**, Second Best: **Black**.

Dataset	Fill Type						
	ZF	**FF**	**BF**	**MeanF**	**MedF**	**ELITS**	**w/o MV**
ETTh1	0.456	0.405	0.407	0.404	0.405	**0.396**	**0.394**
ETTh2	0.558	0.336	0.348	**0.327**	**0.329**	**0.327**	0.331
ETTm1	0.372	**0.341**	0.345	0.347	0.346	0.344	**0.34**
ETTm2	0.277	0.188	0.192	**0.185**	0.187	**0.186**	**0.185**
weather	0.196	**0.178**	0.180	0.19	0.188	**0.178**	**0.177**
electricity	0.232	0.209	0.211	0.223	0.225	**0.195**	**0.189**
traffic	0.618	0.584	0.588	0.619	0.621	**0.538**	**0.526**
% Imp	25.24	3.56	4.85	6.15	6.47	-	-

Ablation

Table 6: Ablation studies. Average MSE on point masking across missing value ratios are reported. Abbreviations - With (w), Without (w/o), Stacked Mixers (**SM**), Preserve D (**PresD**), Inter Patch Mixing in D (**InterD**), Intra Patch Mixing in D (**IntraD**), Channel Mixing (**CM**), Channel Patching (**CP**) and Permute Channels (**PermC**). % Dec denotes the decrement in performance of ELITS when components are either included or excluded.

Dataset	ELITS	w **SM**	w/o **HM**	w/o **IntraD**	w **InterD**	w **PresD**	w/o **CM**	w/o **CP**	w **PermC**
ETTh1	0.049	0.071	0.09	0.054	0.047	0.068	0.111	-	-
ETTh2	0.043	0.046	0.057	0.043	0.042	0.047	0.065	-	-
ETTm1	0.024	0.032	0.039	0.025	0.023	0.031	0.043	-	-
ETTm2	0.021	0.023	0.028	0.021	0.02	0.023	0.029	-	-
weather	0.027	0.029	0.032	0.027	0.028	0.029	0.032	0.028	0.03
electricity	0.051	0.064	0.077	0.054	0.05	0.056	0.069	0.049	0.055
traffic	0.217	0.258	0.283	0.222	0.212	0.238	0.266	0.212	0.223
% Dec	-	20.97	40.32	3.23	-3.23	12.9	41.94	-2.03	4.41

In this section we study the merits of different design choices empirically via ablation. All studies are carried out with point masking and repeated with 3 different seeds on four different missing value ratios (0.125, 0.25, 0.375, and 0.5). The study compares average MSE across seeds and missing value rates.

Mixers In Parallel When mixers are stacked, there is a possibility that information from a preceding mixer might

negatively influence the succeeding ones. This can be eliminated by having mixers in parallel and then combining information from each of the isolated mixers. From Table 6 (Column w **SM**), it can be observed that stacking mixers leads to a degraded performance.

Head Mixer Head Mixer is crucial to combining information from isolated mixers, as one or more types of correlations learnt by the mixers could be vital for the model in order to effectively capture the underlying dynamics of the time series. In Table 6 (Column w/o **HM**), when Head Mixer is removed, there is a drastic drop in the performance across all datasets.

Intra Patch Mixing in D Intra-patch mixers focus on fine-grained local patterns and benefit from higher latent dimensions ($D \approx 2-3\times$ patch size). Inter-patch mixers operate on global patterns and work effectively in data dimension without latent projection. This selective use of latent dimensions balances expressiveness with efficiency. Table 6 (Column w/o **IntraD**) showcases the marginal drop in performance when the latent dimension is not utilized while learning lower level features.

Inter Patch Mixing not in D Inter SPM and Inter CPM learns correlations between patches which focuses on higher level features at the patch level, and can be explored in the data dimension without the need for higher dimensions. From Table 6 (Column w **InterD**), it can be observed that there is no considerable benefit from utilizing the latent dimension to learn higher level features, despite having slightly higher number of parameters.

Reconstruction After Mixing Unlike forecasting where maintaining latent dimension D may capture future information, imputation focuses solely on reconstructing existing values. Projecting back from latent dimension D to data space after mixing reduces parameters while maintaining performance, as the latent space serves only to enrich local feature learning rather than predict future states. Table 6 (Column w **PresD**) confirms this, showing performance drops on most datasets when D is preserved.

Channel Mixing and Channel Patching Channel patching addresses the quadratic parameter growth in traditional mixers. By dividing C channels into N_C patches of size L_C, we reduce the parameter complexity. Mixing within channel patches captures local correlations, while mixing across patches captures global patterns—providing hierarchical channel representations. This design reduces model size significantly on high-dimensional datasets (e.g., Traffic: 862 channels) while improving robustness.

Table 6 shows that disabling channel mixing (w/o **CM**) reduces performance significantly. Disabling channel patching (w/o **CP**) achieves only 2% performance gain while increasing parameters by 8-14× (Table 3), making channel patching a worthwhile trade-off. Channel patching also proves robust to channel ordering, as random permutation (w **PermC**) shows no substantial performance impact. Note that ETT datasets use $L_C = C$ due to having only 7 channels.

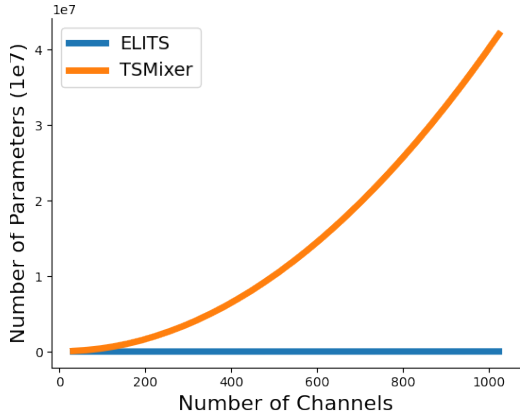


Figure 6: Growth in number of parameters of ELITS when compared against TSMixer. ELITS combats the issue of quadratic parameter growth in traditional mixer models through channel patching, thus number of parameters remains mostly constant as the number of channels increases.

Conclusion

ELITS offers a promising approach to multivariate imputation with enhanced improvements to MLP-Mixer architecture designed for efficiency and accuracy. By incorporating techniques such as Channel Patching, Mixers-in-Parallel, and Head-Mixing, ELITS effectively reduces model complexity while enhancing imputation performance. Unlike traditional methods that often lead to larger models as the number of channels increases, ELITS maintains a more compact size. Our evaluations indicate that ELITS provides a notable improvement in imputation performance while using significantly fewer parameters, breaking the existing trend in literature that larger models often tend to perform better. We believe the parameter and computational efficiency benefits offered by ELITS could lead to widespread adoption of imputation models in resource constrained environments.

References

Adhikari, D.; Jiang, W.; Zhan, J.; He, Z.; Rawat, D. B.; Aickelin, U.; and Khorshidi, H. A. 2022. A comprehensive survey on imputation of missing data in internet of things. *ACM Computing Surveys*, 55(7): 1–38.

Alcaraz, J. M. L.; and Strodthoff, N. 2022. Diffusion-based time series imputation and forecasting with structured state space models. *arXiv preprint arXiv:2208.09399*.

Cao, W.; Wang, D.; Li, J.; Zhou, H.; Li, L.; and Li, Y. 2018. Brits: Bidirectional recurrent imputation for time series. *Advances in neural information processing systems*, 31.

Du, W.; Côté, D.; and Liu, Y. 2023. Saits: Self-attention-based imputation for time series. *Expert Systems with Applications*, 219: 119619.

Ekambaram, V.; Jati, A.; Dayama, P.; Mukherjee, S.; Nguyen, N. H.; Gifford, W. M.; Reddy, C.; and Kalagnanam, J. 2024. Tiny Time Mixers (TTMs): Fast Pre-trained Models for Enhanced Zero/Few-Shot Forecasting of Multivariate Time Series. arXiv:2401.03955.

Ekambaram, V.; Jati, A.; Nguyen, N.; Sinthong, P.; and Kalagnanam, J. 2023. Tsmixer: Lightweight mlp-mixer model for multivariate time series forecasting. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 459–469.

Fang, C.; and Wang, C. 2020. Time Series Data Imputation: A Survey on Deep Learning Approaches. arXiv:2011.11347.

Gao, S.; Koker, T.; Queen, O.; Hartvigsen, T.; Tsiligkaridis, T.; and Zitnik, M. 2024. Units: Building a unified time series model. *arXiv preprint arXiv:2403.00131*.

Goswami, M.; Szafer, K.; Choudhry, A.; Cai, Y.; Li, S.; and Dubrawski, A. 2024. MOMENT: A Family of Open Time-series Foundation Models. *International Conference on Machine Learning (ICML)*.

Khan, H.; Wang, X.; and Liu, H. 2022. Handling missing data through deep convolutional neural network. *Information Sciences*, 595: 278–293.

Kim, T.; Kim, J.; Tae, Y.; Park, C.; Choi, J.-H.; and Choo, J. 2022. Reversible Instance Normalization for Accurate Time-Series Forecasting against Distribution Shift. In *International Conference on Learning Representations*.

Kingma, D. P.; and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kitaev, N.; Kaiser, Ł.; and Levskaya, A. 2020. Reformer: The efficient transformer. *arXiv preprint arXiv:2001.04451*.

Lee, S.; Park, T.; and Lee, K. 2024. Learning to Embed Time Series Patches Independently. In *The Twelfth International Conference on Learning Representations*.

Lin, S.; Lin, W.; Wu, W.; Chen, H.; and Yang, J. 2024. SparseTSF: Modeling Long-term Time Series Forecasting with 1k Parameters. *arXiv preprint arXiv:2405.00946*.

Liu, Y.; Wu, H.; Wang, J.; and Long, M. 2022. Non-stationary Transformers: Exploring the Stationarity in Time Series Forecasting. In Koyejo, S.; Mohamed, S.; Agarwal, A.; Belgrave, D.; Cho, K.; and Oh, A., eds., *Advances in Neural Information Processing Systems*, volume 35, 9881–9893. Curran Associates, Inc.

Loshchilov, I.; and Hutter, F. 2017. SGDR: Stochastic Gradient Descent with Warm Restarts. In *International Conference on Learning Representations*.

Luo, Y.; Cai, X.; Zhang, Y.; Xu, J.; et al. 2018. Multivariate time series imputation with generative adversarial networks. *Advances in neural information processing systems*.

Miao, X.; Wu, Y.; Wang, J.; Gao, Y.; Mao, X.; and Yin, J. 2021. Generative semi-supervised learning for multivariate time series imputation. In *Proceedings of the AAAI conference on artificial intelligence*, 8983–8991.

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

Qin, R.; and Wang, Y. 2023. ImputeGAN: Generative adversarial network for multivariate time series imputation. *Entropy*.

thuml. 2023. Time-Series-Library.

Tolstikhin, I. O.; Houlsby, N.; Kolesnikov, A.; Beyer, L.; Zhai, X.; Unterthiner, T.; Yung, J.; Steiner, A.; Keysers, D.; Uszkoreit, J.; et al. 2021. Mlp-mixer: An all-mlp architecture for vision. *Advances in neural information processing systems*, 34: 24261–24272.

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, S.; Wu, H.; Shi, X.; Hu, T.; Luo, H.; Ma, L.; Zhang, J. Y.; and ZHOU, J. 2024b. TimeMixer: Decomposable Multiscale Mixing for Time Series Forecasting. In *International Conference on Learning Representations (ICLR)*.

Wang, Z.; and Oates, T. 2015. Imaging Time-Series to Improve Classification and Imputation. arXiv:1506.00327.

Woo, G.; Liu, C.; Sahoo, D.; Kumar, A.; and Hoi, S. C. H. 2022. ETSformer: Exponential Smoothing Transformers for Time-series Forecasting. *CoRR*, abs/2202.01381.

Wu, H.; Hu, T.; Liu, Y.; Zhou, H.; Wang, J.; and Long, M. 2023. TimesNet: Temporal 2D-Variation Modeling for General Time Series Analysis. In *International Conference on Learning Representations*.

Wu, H.; Xu, J.; Wang, J.; and Long, M. 2021. Autoformer: Decomposition Transformers with Auto-Correlation for Long-Term Series Forecasting. In *Advances in Neural Information Processing Systems*.

Zeng, A.; Chen, M.; Zhang, L.; and Xu, Q. 2022. Are Transformers Effective for Time Series Forecasting? *arXiv preprint arXiv:2205.13504*.

Zhou, H.; Zhang, S.; Peng, J.; Zhang, S.; Li, J.; Xiong, H.; and Zhang, W. 2021. Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting. In *The Thirty-Fifth AAAI Conference on Artificial Intelligence*, volume 35, 11106–11115.

Zhou, T.; Ma, Z.; Wen, Q.; Wang, X.; Sun, L.; and Jin, R. 2022. FEDformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *Proc. 39th International Conference on Machine Learning*.

Zhou, T.; Niu, P.; Wang, X.; Sun, L.; and Jin, R. 2023. One Fits All: Power General Time Series Analysis by Pretrained LM. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Acknowledgments

This work was generously supported by IBM-IISc Hybrid Cloud Lab collaborative research grant to IISc