MixForecast: Mixer-Enhanced Foundation Model for Load Forecasting

Anuj Kumar
Robert Bosch Centre for Cyber Physical Systems
Indian Institute of Science
Bangalore, India
anujkumar1@alum.iisc.ac.in

Harish Kumar Saravanan
Robert Bosch Centre for Cyber Physical Systems
Indian Institute of Science
Bangalore, India
harishk@iisc.ac.in

Shivam Dwivedi
Robert Bosch Centre for Cyber Physical Systems
Indian Institute of Science
Bangalore, India
dshivam@iisc.ac.in

Pandarasamy Arjunan
Robert Bosch Centre for Cyber Physical Systems
Indian Institute of Science
Bangalore, India
samy@iisc.ac.in

Abstract

Short-term Load Forecasting (STLF) for buildings is essential for optimizing energy management and supporting renewable energy integration, but traditional models often struggle with generalization across diverse building profiles. While recent Time Series Foundation Models (TSFMs) show promise, they remain underexplored for STLF. In this paper, we introduce MixForecast, a novel TSFM for universal energy forecasting. The MixForecast architecture is based on the TSMixer block ensembling technique to provide accurate and robust load forecasting for smart meter time series. In particular, MixForecast is designed with fewer parameters approx 0.19M than existing TSFMs, making it efficient and suitable for deployment in individual buildings. We trained MixForecast using hourly energy data from 63K buildings and evaluated its performance on a test set of 1,000 commercial and residential buildings around the world. We compared MixForecast against pre-trained TSFMs such as Tiny Time Mixers, Lag-Llama, Moirai, and Chronos, in both zero-shot and fine-tuned settings, as well as against traditional models. The model demonstrates superior accuracy and adaptability, excelling in various building profiles. Its lightweight design, combined with strong forecasting performance, establishes MixForecast as a versatile and efficient model for STLF, advancing energy management while promoting sustainable energy practices.

CCS Concepts

• **Computing methodologies** → **Machine learning algorithms**; • **Hardware** → *Energy metering*; *Smart grid*.

Keywords

Energy Informatics, Smart Grid, Energy Forecasting, Short-term Load Forecasting (STLF), Demand-side Load management, Time Series Foundation Models (TSFM), Machine Learning

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).
FMSys '25, May 6–9, 2025, Irvine, CA, USA
© 2025 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-XXXX-X/18/06
https://doi.org/10.1145/3722565.3727193

ACM Reference Format:
Anuj Kumar, Harish Kumar Saravanan, Shivam Dwivedi, and Pandarasamy Arjunan. 2025. MixForecast: Mixer-Enhanced Foundation Model for Load Forecasting. In *The 2nd International Workshop on Foundation Models for Cyber-Physical Systems & Internet of Things (FMSys '25), May 6–9, 2025, Irvine, CA, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3722565.3727193

1 Introduction

Short-term Load Forecasting (STLF) is a challenging task due to fluctuations in energy demand caused by factors like weather, occupancy patterns, and building characteristics. Accurate load forecasting is increasingly important due to the integration of renewable energy sources and rising electricity demand. Traditional forecasting models like ARIMA [7] have limited scalability and struggle to capture complex patterns, while machine learning models, including neural networks and ensemble methods like NBEATS [8], improve accuracy but still struggle with large, diverse datasets.

To address these challenges, we introduce **MixForecast**, a lightweight and scalable model for STLF of smart buildings. MixForecast combines NBEATS and TSMixer [3] blocks, creating an efficient model suitable for edge devices. Trained on real-world hourly energy data from 63,103 buildings, MixForecast is evaluated on 1,000 commercial and residential buildings, outperforming state-of-the-art TSFMs. Importantly, we avoid training on synthetic data to ensure more reliable, generalizable performance.

Recent deep learning models like NBEATS and TSMixer offer scalable solutions for time series analysis, but their applicability to energy load forecasting for smart buildings remains underexplored. This paper presents a comprehensive evaluation of MixForecast alongside TSFMs, including Tiny Time Mixers(TTMs) [2], Chronos [1], Lag-Llama [11], and Moirai [14], in both zero-shot and fine-tuned scenarios. We also compare MixForecast with traditional models like Auto-ARIMA [7], Linear Regression, LightGBM [9]. Our results demonstrate that MixForecast not only outperforms larger TSFMs but is also more efficient and practical for real-world energy management systems.

The key contributions of this paper include:

- **Efficient Load Forecasting:** Introducing MixForecast, a novel model that combines NBEATS and TSMixer blocks for accurate and efficient STLF in smart buildings, optimized for edge-device deployment.
- **Pre-training and Zero-shot Evaluation:** Training MixForecast on real-world data from 63,103 buildings and evaluating its zero-shot performance on 1,000 commercial and residential buildings.
- **Comprehensive Comparison with TSFMs:** Benchmarking MixForecast against state-of-the-art TSFMs, including TTMs, Lag-Llama, Moirai, and Chronos, in both zero-shot and fine-tuned scenarios to show its superior performance.
- **Comparison with Baseline Models:** Demonstrating MixForecast's advantage over traditional models (Auto-ARIMA, Linear Regression, LightGBM) and TFST like TFT [6] in terms of accuracy and adaptability.

2 Related Works

Several traditional ML models have been widely used for load forecasting tasks in the literature. The models include: Auto-ARIMA [7] , an automatically tuned ARIMA model- a popular time series forecasting method based on the autoregressive integrated moving average approach. Linear Regression is a simple yet effective technique for predicting continuous outcomes, and LightGBM [9] is a gradient boosting framework known for its efficiency and scalability, particularly in handling large datasets. Various studies in the literature have applied TSFMs for load demand forecasting. In a study using energy time series dataset, Buildings-900K of NREL [4], transformer based model was used for zero-shot short term load forecasting. IBM Research has introduced a pre-trained TSFM called Tiny Time Mixers (TTMs) [2] for zero-shot and few-shot multivariate time series forecasting.

Recently, a variety of foundation models have been specifically developed for time series data [5]. Based on their inherent architecture, TSFMs can be broadly classified into Transformer-based models, non-Transformer-based models, and diffusion-based models [5]. Transformer-based models leverage the transformer architecture [13], which is known for its ability to handle sequential data effectively. Non-Transformer-based models may use MLP, recurrent neural networks (RNNs) or other statistical methods to analyze time series data, focusing on capturing temporal patterns without the complexity of transformers. Diffusion-based models are a newer class of models that apply diffusion processes to time series data. They aim to model the evolution of time series over time, capturing the dynamics and changes in the data distribution effectively [5]. Lag-Llama [11], a foundational model for univariate probabilistic time series forecasting, was developed by researchers from the Université de Montréal, Mila-Québec AI Institute, and McGill University. Moirai [14], another foundational model for universal forecasting by Salesforce AI Research, is trained on the Large-scale Open Time Series Archive (LOTSA) dataset, containing 27B observations across nine domains. Chronos [1], based on the T5 family [10], was developed by Amazon Science. Some of these models serve as baselines for comparing the performance of MixForecast.

3 Methodology

3.1 Problem Statement

We consider point forecasting problem in discrete time, where given a backcast length bl of 168 (7 days) of hourly data, $\mathbf{y} = [y_1, \ldots, y_{bl}] \in \mathbb{R}^{c \times bl}$, where each $y_i \in \mathbb{R}^{c \times 1}$, the goal is to predict future values $\hat{\mathbf{y}} = [\hat{y}_{bl+1}, \ldots, \hat{y}_{bl+fl}] \in \mathbb{R}^{c \times fl}$, with forecast length fl of 24 (1 day). Here, c represents the number of features: $c = 1$ for univariate and $c > 1$ for multivariate time series. The objective is to minimize forecasting error using **Huber** loss between predicted $\hat{\mathbf{y}}$ and actual values $\mathbf{y}$, evaluated using the **NRMSE** metric.

3.2 MixForecast Model

The **MixForecast** model combines the NBEATS [8] and TSMixer [3] architectures to capture complex temporal patterns while maintaining scalability. By replacing the fully connected layers in NBEATS with TSMixer block, the model improves its ability to understand temporal dependencies, patch-level relationships, and multivariate time series. The feature-level mixing in TSMixer further improves the model's capacity to capture interactions across different features.

The key steps in the model are as follows:

- **Stack Input:** Each stack of the model receives a time series input $\mathbf{x} \in \mathbb{R}^{c \times bl}$, where c is the number of features and bl is the backcast length. Each stack learns one component: trend, seasonality, or error.
- **Block Input:** The L-th block of the model receives a time series input $\mathbf{x}_l \in \mathbb{R}^{c \times bl}$. The input to the next block, $\mathbf{x}_{l+1}$, is the residual $(\mathbf{x}_l - \mathbf{b}_l)$, ensuring that each block learns the previous block's error.
- **TSMixer Block Integration:** The TSMixer block processes time series input with inter-patch, intra-patch, and feature mixing layers to capture both the long-term and the short-term patterns. The L-th TSMixer block transformation is:
$$\mathbf{h}_l = \mathbf{TSMixer}(\mathbf{x}_l) \in \mathbb{R}^{c \times bl}.$$
- **Backcast and Forecast Components:** The output of the L-th TSMixer block, $\mathbf{h}_l$, is projected onto the backcast Θ_l^b and forecast Θ_l^f components:
$$\Theta_l^b = \mathbf{W}_l^b \mathbf{h}_l \in \mathbb{R}^{c \times p}, \quad \Theta_l^f = \mathbf{W}_l^f \mathbf{h}_l \in \mathbb{R}^{c \times q}.$$
The sizes of $p, q \lll bl$, and $\mathbf{W}_l^b$,$\mathbf{W}_l^f$ are learnable parameters.
- **Backcast and Forecast Outputs:** These components are passed through learnable or non-learnable functions $\mathbf{g}_l^b$ and $\mathbf{g}_l^f$ to capture trends and seasonalities:
$$\mathbf{b}_l = \mathbf{g}_l^b(\Theta_l^b) \in \mathbb{R}^{c \times bl}, \quad \mathbf{f}_l = \mathbf{g}_l^f(\Theta_l^f) \in \mathbb{R}^{c \times fl}.$$
For the L-th block, $\mathbf{b}_l$ represents the approximation of the input block, while $\mathbf{f}_l$ denotes the prediction from this block.
- **Global Forecast:** The global forecast is the aggregation of all forecast outputs across blocks:
$$\mathbf{F} = \sum_{l=1}^{L} \mathbf{f}_l,$$

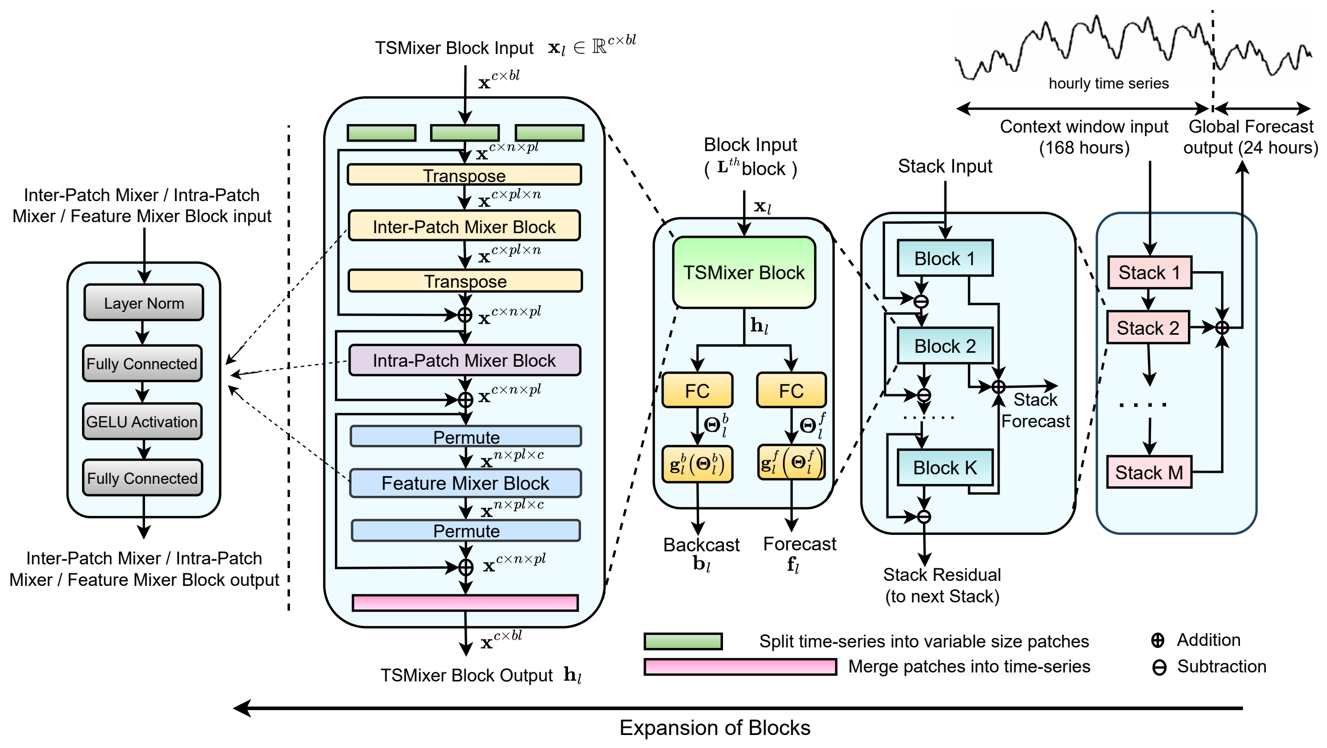
TSMixer Block Input $\mathbf{x}_l \in \mathbb{R}^{c\times bl}$
Inter-Patch Mixer / Intra-Patch Mixer / Feature Mixer Block input
Layer Norm
Fully Connected
GELU Activation
Fully Connected
Inter-Patch Mixer / Intra-Patch Mixer / Feature Mixer Block output
Transpose
Inter-Patch Mixer Block
Intra-Patch Mixer Block
Permute
Feature Mixer Block
TSMixer Block Output $\mathbf{h}_l$
Block Input ($\mathbf{L}^{th}$ block)
TSMixer Block
FC
Backcast $\mathbf{b}_l$
Forecast $\mathbf{f}_l$
Stack Input
Block 1
Block 2
Block K
Stack Forecast
Stack Residual (to next Stack)
hourly time series
Context window input (168 hours)
Global Forecast output (24 hours)
Stack 1
Stack 2
Stack M
Split time-series into variable size patches
Merge patches into time-series
Addition
Subtraction
Expansion of Blocks

Figure 1: Architecture of MixForecast.

where $L = M \times K$ is the total number of blocks, with M as the number of stacks and K as blocks per stack.

This hybrid architecture, combining NBEATS and TSMixer, offers an efficient, scalable model for Short-term Load Forecasting (STLF), as shown in Figure 1.

3.2.1 *Modified NBEATS Block.*

The NBEATS model consists of multiple blocks, each capturing different temporal patterns (trend and seasonality). The block output is split into backcast (reconstructed input) and forecast (predicted future values) components via projection layers. For a block, the forward pass is given by:

$$\Theta^b = \mathbf{W}^b\mathbf{h}, \quad \Theta^f = \mathbf{W}^f\mathbf{h}, \tag{1}$$

$$\mathbf{b} = \mathbf{g}^b(\Theta^b) \in \mathbb{R}^{c\times bl}, \quad \mathbf{f} = \mathbf{g}^f(\Theta^f) \in \mathbb{R}^{c\times fl}, \tag{2}$$

where $\mathbf{h}$ is one of TSMixer block output, and $\mathbf{W}^b, \mathbf{W}^f$ are learnable parameters for backcast and forecast respectively and the functions $\mathbf{g}^b$ and $\mathbf{g}^f$ capture trends and seasonalities.

3.2.2 *Modified TSMixer Block.*

TSMixer block [3], inspired by MLP-Mixer [12], processes time series in a patch-based manner. The input $\mathbf{x} \in \mathbb{R}^{c\times bl}$ is divided into n patches of length pl, where $bl = n \times pl$. The number of patches n is adjusted by a factor k as $n' = n\times k$, modifying patch length to $pl' = \frac{pl}{k}$. Each block processes patches of varying sizes to capture information at different scales of seasonality, allowing the model to learn diverse temporal patterns across the time series. TSMixer block consists of three components:

(1) **Inter-Patch Mixer:** Captures interactions between patches. Transforming input $\mathbf{x} \in \mathbb{R}^{c\times n\times pl}$ as Transform$(\mathbf{x}) \in \mathbb{R}^{c\times pl\times n}$, inter-patch mixing is performed by:

$$\mathbf{x}' = \text{LayerNorm}(\text{Transform}(\mathbf{x}))\mathbf{W}^n, \tag{3}$$

$$\mathbf{x} = \mathbf{x} + \text{Transform}(\text{GELU}(\mathbf{x}')\mathbf{W}'^n). \tag{4}$$

(2) **Intra-Patch Mixer:** Captures intra-patch interactions. After inter-patch mixing, intra-patch mixing is performed as:

$$\mathbf{x}' = \text{LayerNorm}(\mathbf{x})\mathbf{W}^{pl}, \tag{5}$$

$$\mathbf{x} = \mathbf{x} + \text{GELU}(\mathbf{x}')\mathbf{W}'^{pl}. \tag{6}$$

(3) **Feature Mixer:** Captures feature-level interactions across time steps. Permuting input $\mathbf{x} \in \mathbb{R}^{c\times n\times pl}$ as Permute$(\mathbf{x}) \in \mathbb{R}^{n\times pl\times c}$, feature mixing is performed as:

$$\mathbf{x}' = \text{LayerNorm}(\text{Permute}(\mathbf{x}))\mathbf{W}^c, \tag{7}$$

$$\mathbf{x} = \mathbf{x} + \text{Permute}(\text{GELU}(\mathbf{x}')\mathbf{W}'^c). \tag{8}$$

Each block captures temporal and patch-level relationships, enabling model to handle long-term dependencies effectively. Here, $\mathbf{W}^n, \mathbf{W}'^n, \mathbf{W}^{pl}, \mathbf{W}'^{pl}, \mathbf{W}^c, \mathbf{W}'^c$, are learnable parameters.

4 Experiments and Results

4.1 Datasets

As part of this study, we collected energy datasets from 64,103 real-world buildings, encompassing both commercial and residential categories across various countries worldwide. It is to be noted

Table 1: Comparison of model attributes.

Model	Architecture	Parameters	Max. Backcast Length	Pre-training Data (Size)	Forecasting type
Lag-Llama [11]	Decoder-only	2.45M	1024	1B observations	Probabilistic
Moirai [14]	Encoder-only	14M	5000	27B observations	Probabilistic
Chronos [1]	Encoder-Decoder	46M	512	893K series	Probabilistic
NBEATS [8]	MLP	2.2M	240	282K series	Point
TTMs [2]	MLP	1M	1536	282K series	Point
MixForecast	MLP	**0.19M**	168	63K series	Point

Table 2: Details of the real building datasets used in this study.

Datasets	Location	No. of Buildings	Year Range
Commercial Buildings			
Enernoc	USA	100	2012
IBlend	India	9	2013-17
DGS	USA	322	2016-2018
EWELD	China	386	2016-2022
Residential Buildings			
CEEW	India	84	2019-21
Ireland	Ireland	20	2020
MFRED	USA	26	2019
NEEA	USA	192	2018-20
NEST	Switzerland	1	2019-23
Prayas	India	116	2018-20
SMART*	USA	114	2016
SGSC	Australia	13,735	2011-14
GoiEner	Spain	25,559	2014-22
UK SMART	UK	14,319	2008-2010
SAVE	UK	4691	2016-2018
iFlex	Norway	4429	2020-2021
Total		**64,103**	

that, while other public synthetic datasets are available, we excluded them from this study as they have already been used in one or more existing TSFMs for pre-training (see Table 1). Including such datasets in our analysis would lead to biased results. Table 2 provides a comparison of these datasets.

4.2 Implementation and Setup

Our experimental setup uses JupyterLab for development and the GluonTS[1] (for Lag-Llama and Moirai), AutoGluon[2] (For Chronos and Auto-ARIMA), skforecast (for LightGBM and Linear Regression) and neuralforecast[3] (for Trained from Scratch Transformer) libraries to implement baseline models for evaluating performance, running on a server equipped with 2 GeForce RTX 4090 GPUs. (Note: GPU is optional for Zero-shot forecasting). Our reproducible code repository contains brief details of the collected datasets, results, plots and model error analysis.[4]

[1]https://ts.gluon.ai/stable/index.html
[2]https://auto.gluon.ai/
[3]https://nixtlaverse.nixtla.io/neuralforecast/docs/getting-started/introduction.html
[4]https://github.com/AI-IoT-Lab/mix-forecast

4.3 Sliding Window Extraction

We began by extracting sliding windows for each building. Specifically, we employed an 8-day sliding window comprising a 192-hour load sub-sequence, with a stride of one day. The initial 7 days (168 hourly energy meter readings) served as context to forecast the subsequent 24-hour readings of the 8th day.

4.4 Pre-training

We pretrain model using energy data from 63,103 buildings with a variate patch size of 6, 12, and 24 to efficiently capture temporal patterns. The model consists of 3 stacks ($M = 3$), each containing 3 TSMixer blocks ($K = 3$), with both patch and feature hidden dimensions set to 256. The GELU activation function is used for improved stability and performance. Model is trained for 100 epochs with early stopping to prevent overfitting. Compared to the training of TTMs and NBEATS, our base model requires less training time, approximately 72 hours on 2 GeForce RTX 4090 GPUs.

4.5 Zero-shot

For Zero-shot STLF, MixForecast generates day-ahead hourly forecasts for 1000 unseen commercial and residential buildings, utilizing 168 hours of weekly data for context and a 24-hour forecast. This task evaluates the model's ability to forecast energy consumption for new buildings with minimal historical data.

4.6 Fine-tuning

Fine-tuning adapts a pre-trained TSFM to individual target buildings using one year of hourly data, where the first six months are for training and validation, while the remaining six months are reserved for testing, using only one week's hourly data (168 hours of context and 24 hours forecast). We train the model for up to 10 epochs with a learning rate of 1e-3, employing early stopping to prevent overfitting.

4.7 Evaluation Metric

We assess forecasting performance using Normalized Root Mean Square Error (NRMSE), a standard metric in time series forecasting, also used in BuildingsBench [4]. NRMSE(coefficient of variation of RMSE), capturing the accuracy of load shape prediction, is defined for a target building with M days of energy load time series as:

$$NRMSE = 100 \times \frac{1}{\bar{y}} \sqrt{\frac{1}{24M} \sum_{j=1,i=1}^{M,24} (y_{i,j} - \hat{y}_{i,j})^2}, \quad (9)$$

Table 3: Comparison of the performance of Pre-training from Scratch, TFST, and Traditional models using median NRMSE across different datasets. (TTMs - Tiny Time Mixers, TFST - Trained from Scratch Transformer, TFT - Temporal Fusion Transformer) (Bold - Best, Underlined - 2^{nd} Best)

Dataset	Pre-training from Scratch			TFST	Traditional models		
	NBEATS	TTMs	**MixForecast** (Proposed)	TFT	ARIMA	Linear Regression	LightGBM
Commercial Buildings							
Enernoc	28.60	23.21	22.54	37.92	51.53	28.72	27.13
Iblend	23.00	21.51	19.58	29.49	25.02	26.97	25.97
Average	25.80	<u>22.36</u>	**21.06**	33.71	38.28	27.85	26.55
Residential Buildings							
Mathura	141.72	136.06	132.09	141.60	104.95	101.46	104.76
Bareilly	65.92	64.42	63.28	75.33	78.07	76.68	76.48
MFRED	26.46	24.74	24.39	34.64	42.03	34.15	32.94
NEEA	74.51	69.31	68.52	83.74	92.68	88.07	86.60
NEST	64.95	64.30	64.71	69.64	75.62	68.27	66.18
Prayas	106.85	101.97	100.17	124.09	103.87	81.98	76.25
Smart*	62.75	61.54	61.01	66.42	69.27	93.18	89.10
Ireland	83.45	81.52	81.02	90.11	95.11	95.99	92.29
GoiEner	120.99	117.69	118.27	141.01	119.95	119.20	115.82
SGSC	94.98	92.90	91.47	117.96	111.91	104.22	102.79
Average	84.26	<u>81.45</u>	**80.49**	94.45	89.35	86.32	84.32

Table 4: Comparison of the performance of Zero-shot and Fine-tuned models using median NRMSE across different datasets with a backcast length of 168 and a forecast length of 24. (Bold - Best, Underlined - 2^{nd} Best)

Dataset	Zero-shot				Fine-tuned		
	Moirai	Chronos	Lag-Llama	**MixForecast**	Moirai	Lag-Llama	**MixForecast**
Commercial Buildings							
Enernoc	29.20	25.99	51.68	22.54	24.68	27.98	20.99
Iblend	27.30	16.17	63.16	19.58	22.00	18.44	19.89
Average	28.25	<u>21.08</u>	57.42	**21.06**	23.34	<u>23.21</u>	**20.44**
Residential Buildings							
Mathura	109.60	102.78	113.49	132.09	91.46	99.34	120.97
Bareilly	69.90	73.84	89.88	63.28	54.41	69.27	58.23
MFRED	27.10	25.39	61.97	24.39	20.80	30.81	24.19
NEEA	80.40	82.64	91.93	68.52	66.81	84.27	70.06
NEST	71.70	71.33	85.00	64.71	54.98	66.90	50.54
Prayas	90.40	87.93	100.47	100.17	56.83	78.03	70.91
Smart*	65.90	69.69	83.61	60.01	65.14	89.12	73.97
Ireland	86.50	92.32	115.63	81.02	70.01	82.84	76.63
GoiEner	111.62	114.59	131.23	118.27	99.34	111.07	96.35
SGSC	92.01	99.43	111.91	91.47	85.24	89.23	82.69
Average	<u>80.51</u>	81.99	98.51	**80.49**	**66.50**	80.09	<u>72.45</u>

where $\hat{y}$ is the predicted load, y is the actual load, and $\bar{y}$ is the average actual load over all M days. This metric provides an effective measure of model performance, focusing on the accuracy of the model's load shape predictions.

4.8 Results and Analysis

4.8.1 ***Zero-shot Results:*** In the Zero-shot STLF scenario, pre-trained models are evaluated for day-ahead forecasting on all test buildings without fine-tuning. Tables 3 and 4 compare the performance of MixForecast with TSFMs, TFST, and Traditional models across both commercial and residential datasets. MixForecast outperforms all TSFMs, TFST, and Traditional models for both commercial and residential buildings. Figure 2 illustrates MixForecast's load forecasting for a building from the Enernoc dataset.

4.8.2 ***Fine-tuned Results:*** For fine-tuning, TSFMs Moirai and Lag-Llama were adapted as described in Section 4.6. Table 4 compares the results of fine-tuned MixForecast, Moirai, and Lag-Llama models. Fine-tuning on limited data and fewer epochs significantly improves performance. MixForecast performs better than fine-tuned TSFMs for commercial buildings, while fine-tuning provides a notable improvement over MixForecast for residential buildings.

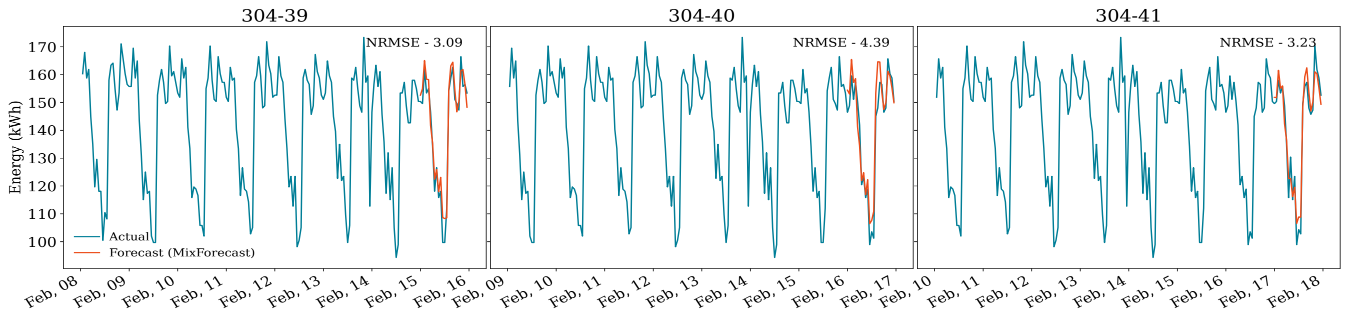

Figure 2: Load forecasting by MixForecast for subsequent windows for building name 304 of Enernoc commercial dataset with NRMSE value for the each window. (Building name-Window number e.g. 304-39)

*4.8.3 **Observations of Forecasting Errors for Commercial Buildings:*** Among Pre-training from Scratch models, MixForecast achieved the lowest NRMSE of 21.06, followed by TTMs with 22.36, outperforming all TSFMs, TFST, and Traditional models. Overall, MixForecast had the lowest NRMSE of 21.06, with Chronos at 21.08. MixForecast also showed a significantly lower NRMSE than Traditional models and TFST. Among fine-tuned TSFMs, MixForecast again led with an NRMSE of 20.44. Notably, Lag-Llama demonstrated a 59.57% error reduction, highlighting the benefit of fine-tuning TSFM on target buildings.

*4.8.4 **Observations of Forecasting Errors for Residential Buildings:*** Among Pre-training from Scratch models, MixForecast achieved the lowest NRMSE of 80.49, followed by TTMs with 81.45, outperforming Traditional models. Among TSFMs, MixForecast had the lowest NRMSE of 80.49, slightly ahead of Moirai at 80.51, and showed significantly lower NRMSE compared to Pre-training from Scratch models, TFST, and Traditional models. In fine-tuned TSFMs, Moirai achieved the lowest NRMSE of 66.50. Moirai, Lag-Llama, and MixForecast show error reductions of 17.40%, 18.70%, and 9.99%, respectively, with smaller reductions in residential buildings compared to commercial ones. The higher error in residential buildings is attributed to greater variability in energy loads, influenced by occupant behavior and environmental factors like temperature and humidity [4].

5 Conclusion and Future Work

We propose TSFM **MixForecast**, a TSMixer block-based ensemble for Short-Term Load Forecasting (STLF) in smart buildings. MixForecast achieves the lowest NRMSE of **20.44** for commercial buildings. In residential buildings, fine-tuning TSFM Moirai resulted in an NRMSE of **66.50**, while MixForecast reached **72.45**, highlighting the impact of fine-tuning. MixForecast also demonstrates high computational efficiency with an inference time of **0.18** seconds per building and a lightweight architecture of **0.19M** parameters, making it suitable for edge devices.

Future work will focus on enhancing MixForecast with larger, multi-resolution datasets for better generalization across domains like traffic flow and healthcare. We also plan to extend it for variable-length predictions, classification, anomaly detection, imputation, and optimize it for long-term forecasting using hybrid approaches. Additionally, real-time deployment for smart grids and residential energy management will be explored.

Acknowledgments

This work was supported by a research grant from the AI & Robotics Technology Park (ARTPARK) at the Indian Institute of Science. The authors gratefully acknowledge the dataset providers for their efforts in making the data publicly available and maintaining high-quality resources for this study.

References

[1] Abdul Fatir Ansari, Lorenzo Stella, Caner Turkmen, Xiyuan Zhang, Pedro Mercado, Huibin Shen, Oleksandr Shchur, Syama Rangapuram, Sebastian Pineda Arango, Shubham Kapoor, et al. 2024. Chronos: Learning the language of time series.

[2] Vijay Ekambaram, Arindam Jati, Pankaj Dayama, Sumanta Mukherjee, Nam H. Nguyen, Wesley M. Gifford, Chandra Reddy, and Jayant Kalagnanam. 2024. Tiny Time Mixers (TTMs): Fast Pre-trained Models for Enhanced Zero/Few-Shot Forecasting of Multivariate Time Series. arXiv:2401.03955 [cs.LG] https://arxiv.org/abs/2401.03955

[3] Vijay Ekambaram, Arindam Jati, Nam Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. 2023. Tsmixer: Lightweight mlp-mixer model for multivariate time series forecasting. , 459–469 pages.

[4] Patrick Emami, Abhijeet Sahu, and Peter Graf. 2023. Buildingsbench: A large-scale dataset of 900k buildings and benchmark for short-term load forecasting. *Advances in Neural Information Processing Systems* 36 (2023), 19823–19857.

[5] Yuxuan Liang, Haomin Wen, Yuqi Nie, Yushan Jiang, Ming Jin, Dongjin Song, Shirui Pan, and Qingsong Wen. 2024. Foundation models for time series analysis: A tutorial and survey.

[6] Bryan Lim, Sercan O. Arik, Nicolas Loeff, and Tomas Pfister. 2020. Temporal Fusion Transformers for Interpretable Multi-horizon Time Series Forecasting. arXiv:1912.09363 [stat.ML] https://arxiv.org/abs/1912.09363

[7] Norizan Mohamed, Maizah Hura Ahmad, Zuhaimy Ismail, and S Suhartono. 2010. Short term load forecasting using double seasonal ARIMA model. , 57–73 pages.

[8] Boris N Oreshkin, Dmitri Carpov, Nicolas Chapados, and Yoshua Bengio. 2019. N-BEATS: Neural basis expansion analysis for interpretable time series forecasting.

[9] Sungwoo Park, Seungmin Jung, Seungwon Jung, Seungmin Rho, and Eenjun Hwang. 2021. Sliding window-based LightGBM model for electric load forecasting using anomaly repair. *J. Supercomput.* 77, 11 (Nov. 2021), 12857–12878. https://doi.org/10.1007/s11227-021-03787-4

[10] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research* 21, 140 (2020), 1–67.

[11] Kashif Rasul, Arjun Ashok, Andrew Robert Williams, Hena Ghonia, Rishika Bhagwatkar, Arian Khorasani, M Javad Darvishi Bayazi, George Adamopoulos, Roland Riachi, Nadhir Hassen, et al. 2023. Lag-llama: Towards foundation models for probabilistic time series forecasting.

[12] Ilya O Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, et al. 2021. Mlp-mixer: An all-mlp architecture for vision. *Advances in neural information processing systems* 34 (2021), 24261–24272.

[13] Ashish Vaswani. 2017. Attention is all you need.

[14] Gerald Woo, Chenghao Liu, Akshat Kumar, Caiming Xiong, Silvio Savarese, and Doyen Sahoo. 2024. Unified training of universal time series forecasting transformers.