Linear vs Transformer Models for Long-Horizon Exogenous Temperature Forecasting

Ruslan Gokhman
Yeshiva University, New York, USA
rgokhman@yu.edu

Abstract

We study long-horizon temperature forecasting using linear and Transformer-family models. We evaluate Linear, NLinear, DLinear, Transformer, Informer, and Autoformer under standardized train, validation, and test splits. Results show that linear baselines (Linear, NLinear, DLinear) consistently outperform more complex Transformer-family architectures, with NLinear achieving the best overall accuracy across all splits. These findings highlight that carefully designed linear models remain strong baselines for time series forecasting in challenging settings.

1 Introduction

Forecasting indoor temperature is vital for smart building management, energy optimization, and occupant comfort. The **UrbanAI 2025 Challenge** offers a stringent testbed: models must forecast long horizons mirroring production scenarios where ground-truth temperatures are unavailable at inference time.

This paper conducts a head-to-head comparison between strong linear baselines—*Linear*, *NLinear*, and *DLinear*—and widely used Transformer-family approaches (Vanilla Transformer, Informer, Autoformer). Our goal is to assess whether the additional capacity and inductive biases of attention-based architectures translate into better *generalization* under the challenge's constraints, or whether carefully constructed linear models remain more reliable.

To ensure fairness, we mirror the LTSF-Linear evaluation protocol where applicable (consistent input/output horizons, basic preprocessing, and standardized metrics). We use identical train/validation/test splits for all methods and report MAE/MSE on the official blind test set. The central question we answer is: *Do Linear, NLinear, and DLinear outperform Transformer-family models in this long-horizon regime, and by what margin?*

2 Dataset and Task Description

Our comparison protocol follows the widely used LTSF-Linear suite [Lab, 2022], which standardizes sequence lengths, splits, and evaluation for linear baselines (Linear, NLinear, DLinear) against Transformer-family models. We mirror those settings where applicable to enable direct comparison.

The dataset originates from the *Smart Buildings Control Suite (SBCS)* [Goldfeder et al., 2025], which provides diverse benchmarks for evaluating HVAC and temperature control policies. The original dataset was provided as a single block and split as follows: training (36,105 samples), validation (10,275 samples) for early stopping only, and an *Official Test* split. Intended to be larger, only 10,563 samples were successfully evaluated due to technical issues. The official test set was reserved for final, blind evaluation, and its ground-truth labels were never accessed during model design or tuning.

Preprint.

Contest rules and evaluation. Participants predict the full temperature sequence for the validation period using only exogenous data; direct or indirect use of validation temperatures during training is forbidden. Submissions may be point predictions, histograms, or mean/std per step. MAE is the main metric for point/histogram outputs; KL divergence is used for mean/std submissions. Evaluations prioritize duration, accuracy, novelty, and reproducibility.

Challenges. Long-term prediction over months inputs, multi-scale patterns (daily/weekly cycles, holidays, regime changes), and potential data gaps.

3 Methods

Baseline suite and comparison. We adopt the configuration spirit of LTSF-Linear [Lab, 2022] to ensure apples-to-apples comparisons across linear and Transformer-family models (same horizons, basic preprocessing, and standardized metrics). Where our challenge rules differ we adapt the dataloaders accordingly.

We explore three competitive linear baselines for long sequence forecasting, and three Transformer-family baselines widely used in time-series.

3.1 Transformer Baseline (Vanilla Transformer)

We implement the encoder–decoder architecture with multi-head self-attention and position-wise feed-forward layers [Vaswani et al., 2017]. We adopt sinusoidal positional encodings, layer normalization, dropout, and teacher-forced direct multi-step decoding (96-step horizon). Input features are the exogenous variables; the decoder receives start tokens plus time features. We tune $d_{\text{model}} \in \{256, 512\}$, heads $\in \{4, 8\}$, encoder/decoder layers $\in \{2, 3\}$, and dropout $\in [0.05, 0.2]$.

3.2 Informer

Informer introduces ProbSparse self-attention to reduce complexity for long sequences and a distillation operation across layers to keep only salient temporal information [Zhou et al., 2021]. We follow the authors' settings for sequence lengths (input 96, output 96), factor $\in \{3, 5\}$, and use the encoder-only forecasting head with generative decoder.

3.3 Autoformer

Autoformer replaces dot-product attention with an auto-correlation mechanism to capture periodic dependencies and includes a seasonal-trend decomposition inside the network [Wu et al., 2021]. We adopt similar hyperparameters as Informer and the original paper, including decomposition kernel sizes in $\{3, 5, 7\}$.

3.4 Linear: Basic Temporal Linear Model

The Linear model from the LTSF-Linear suite is a one-layer temporal projection: it directly maps the input sequence to the forecast horizon with a single linear layer along the time dimension. Despite its simplicity, it has been shown to outperform several popular Transformer-based methods, serving as a surprisingly strong and interpretable benchmark.

3.5 NLinear: Normalized Linear Forecasting

Given an input window $X \in \mathbb{R}^{L \times C}$ with last timestep x_L, NLinear centers the window via $X' = X - x_L$. A learnable linear projection W maps to future predictions, which are then de-normalized: $\hat{X} = WX' + x_L$. This centering improves robustness to nonstationarity.

3.6 DLinear: Decomposed Linear Forecasting

DLinear decomposes the series into trend and seasonal components, $X = X_{\text{trend}} + X_{\text{seasonal}}$. A moving average estimates the trend; each component is linearly forecasted: $\hat{X} = W_{\text{trend}} X_{\text{trend}} + W_{\text{seasonal}} X_{\text{seasonal}}$, capturing long-term drift and periodicity explicitly.

3.7 Implementation Details

All models are implemented in PyTorch with early stopping on validation loss. No external data or augmentation is used. Input and output lengths are 96 steps (e.g., four days). Each prediction uses 96 consecutive exogenous points (weather, setpoints) to forecast the next 96 temperatures, matching the direct multi-step setup. Feature normalization is per-variable using training-set statistics. **All experiments were conducted on a Google Colab environment using an NVIDIA T4 GPU.**

4 Evaluation Metrics

Reproducibility notes for new baselines. For all Transformer-family models, we use Adam with learning rate 1×10^{-4} or 5×10^{-4}, weight decay 1×10^{-4}, batch size $\in \{16, 32\}$, and early stopping on validation MAE with patience 3. We sweep hidden dimension, layers, heads/factor, and dropout; all runs fix input/output horizons to 96/96 for comparability.

We report mean absolute error (MAE) and mean squared error (MSE):

$$\mathrm{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|, \qquad \mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2. \tag{1}$$

For probabilistic predictions, we additionally report KL divergence between predicted and true distributions. All metrics (MAE and MSE) are computed in the normalized feature space. Specifically, each variable is standardized using training-set statistics (z-score normalization), following the LTSF-Linear evaluation protocol. Thus, the reported errors reflect performance in normalized units rather than physical temperature units. This ensures fair and scale-consistent comparison across all baseline models.

5 Results

Table 1 summarizes performance across splits. The *Official Test* results represent the contest-required benchmark. Due to technical constraints, only 10,563 test samples were evaluated (the official test set is larger).

Table 1: Performance of Linear, NLinear, and DLinear. Official test results in **bold**.

Model	Split	Size	MAE	MSE
NLinear	Training	36,105	–	0.0619
DLinear	Training	36,105	–	0.0660
Linear	Training	36,105	–	0.0679
Transformer	Training	36,105	–	0.0330
Informer	Training	36,105	–	–
Autoformer	Training	36,105	–	0.0844
NLinear	Validation	10,275	0.2529	0.4665
DLinear	Validation	10,275	0.2784	0.4744
Linear	Validation	10,275	0.2843	0.4882
Transformer	Validation	10,275	0.3375	0.5302
Informer	Validation	10,275	–	–
Autoformer	Validation	10,275	0.3493	0.6201
NLinear	**Official Test**	**10,563**	**0.2461**	**0.5220**
DLinear	Official Test	10,563	0.2811	0.5489
Linear	Official Test	10,563	0.2862	0.5634
Transformer	Official Test	10,563	0.8342	1.4371
Informer	Official Test	10,563	0.8342	1.4371
Autoformer	Official Test	10,563	0.3598	0.7368

6 Discussion

Linear baselines (Linear, NLinear, DLinear) remain the strongest performers across all splits. NLinear achieved the best overall accuracy, with low training error, the strongest validation MAE/MSE, and the best test performance (MAE 0.2461, MSE 0.5220). DLinear followed closely but consistently underperformed NLinear by a small margin. Linear produced slightly weaker results than NLinear/DLinear but still significantly outperformed Transformer-family models on the test set (MAE 0.2862, MSE 0.5634). Transformer achieved strong training/validation performance (MSE 0.0330 / 0.5302) but collapsed on the test set (MSE 1.4371), demonstrating poor generalization in the setting. Autoformer produced moderate results, better than Transformer/Informer on the test set but worse than linear baselines. Informer underperformed across the board, matching Transformer's weak test metrics.

These findings highlight that despite advances in attention-based architectures, simple and interpretable linear models remain robust and competitive for long-horizon time series forecasting.

7 Conclusion

Carefully designed linear models (Linear, NLinear, DLinear) provide strong baselines for the long-horizon temperature forecasting in smart buildings, combining efficiency, interpretability, and competitive accuracy. As datasets grow, integrating modest nonlinearity and leveraging transfer learning are promising next steps.

Broader impacts. Improved temperature forecasting can aid energy savings, carbon reduction, and comfort at scale. Risks include overreliance on models without human oversight; we recommend monitoring, calibration checks, and transparent reporting of uncertainty.

Acknowledgments and Disclosure of Funding

Use unnumbered first-level headings. Place funding and competing interest disclosures here. Do **not** include this section in the anonymized submission; it will be hidden automatically by the style's `ack` environment in submission mode.

References

Judah Goldfeder, Victoria Dean, Zixin Jiang, Xuezheng Wang, Bing Dong, Hod Lipson, and John Sipple. The smart buildings control suite: A diverse open source benchmark to evaluate and scale hvac control policies for sustainability. *arXiv preprint arXiv:2410.03756*, 2025. URL `https://arxiv.org/abs/2410.03756`.

CURE Lab. Ltsf-linear: Linear baselines for long-term time series forecasting. `https://github.com/cure-lab/LTSF-Linear`, 2022.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, 2017.

Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. In *Advances in Neural Information Processing Systems*, 2021.

Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?
   Answer: [Yes]
   Justification: The claims (linear baselines, official test MAE/MSE) match the reported methods and results (Secs. 1).

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?
   Answer: [Yes]
   Justification: We note linearity constraints, lack of nonlinearity, and partial test-set evaluation due to technical issues (Secs. Methods, Results, Discussion).

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete proof?
   Answer: [NA]
   Justification: This work is empirical; no new theorems are introduced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all information needed to reproduce the main experimental results?
   Answer: [Yes]
   Justification: We specify model types, horizons, optimizer, batch sizes, normalization, and exact metrics; code details can be added in the supplemental.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code with sufficient instructions?
   Answer: [No]
   Justification: Dataset is from a public challenge; we will release scripts and instructions in anonymized supplemental upon submission.

6. **Experimental setting/details**

   Question: Are all training/test details specified?
   Answer: [Yes]
   Justification: Horizons, splits, training hyperparameters, and metrics are reported; supplemental will include full configs.

7. **Experiment statistical significance**

   Question: Are error bars/significance or similar appropriate statistics reported?
   Answer: [No]
   Justification: We report point metrics for the main benchmark; future work will add variability across seeds/runs.

8. **Experiments compute resources**

   Question: Are compute resources disclosed?
   Answer: [Yes]
   Justification: All experiments were conducted on a Google Colab environment using an NVIDIA T4 GPU; this setup was sufficient to train and evaluate the considered models.

9. **Code of ethics**

   Question: Does the work conform to the NeurIPS Code of Ethics?
   Answer: [Yes]
   Justification: The study uses publicly available benchmark data and standard evaluation protocols.

10. **Broader impacts**

Question: Are positive and negative societal impacts discussed?
Answer: [Yes]
Justification: We outline benefits (energy savings) and risks (overreliance without oversight) with suggested mitigations.

11. **Safeguards**

    Question: Are safeguards described for high-risk assets?
    Answer: [NA]
    Justification: No high-risk models/datasets are released; methods are simple linear predictors.

12. **Licenses for existing assets**

    Question: Are licenses/terms for third-party assets documented and respected?
    Answer: [Yes]
    Justification: We cite dataset and prior work; licenses/terms will be documented in the supplemental and repository.

13. **New assets**

    Question: Are new assets documented?
    Answer: [No]
    Justification: No new datasets or pretrained models are released in this work.

14. **Crowdsourcing and human subjects**

    Question: For crowdsourcing/human-subjects research, are instructions and compensation details included?
    Answer: [NA]
    Justification: Not applicable.

15. **IRB approvals**

    Question: Are risks/IRB approvals described?
    Answer: [NA]
    Justification: Not applicable.

16. **Declaration of LLM usage**

    Question: Is LLM usage in core methods described if relevant?
    Answer: [No]
    Justification: No LLMs are used in the core methods; any writing assistance will be disclosed separately if required.