ModernTCN Revisited: A Reproducibility Study with Extended Benchmarks and Architectural Improvement

Anonymous authors
Paper under double-blind review

Abstract

This study revisits ModernTCN, a temporal convolutional network designed for time series analysis, to evaluate its performance claims and extend its applicability through architectural innovation. We conduct a comprehensive evaluation across various time series tasks, addressing methodological issues such as the "Drop Last Trick" and comparing ModernTCN against a broader set of benchmarks, including global convolutional models. Our findings reveal that while ModernTCN demonstrates competitive performance, it is sensitive to experimental setups and data handling, particularly in the long-term forecasting task. By integrating ModernTCN with a global continuous kernel convolutional method, we achieve state-of-the-art performance on the PhysioNet 2019 dataset, highlighting the potential of this innovative combination for irregularly sampled data. Additionally, our analysis of effective receptive fields (ERFs) and parameter efficiency underscores the trade-offs inherent in ModernTCN's design, particularly its maintenance of an extra variable dimension. The cross-variable component (ConvFFN2) is validated as crucial for handling missing data, aligning with the original study's findings. This work not only reproduces and extends the benchmarks for ModernTCN but also provides insights into its architectural strengths and limitations, paving the way for future research in time series modeling.

1 Introduction

Time series analysis is a fundamental problem with broad applications across various domains, including weather forecasting (Bi et al., 2023), anomaly detection in spacecraft monitoring (Su et al., 2019b), medical symptom classification (Kiyasseh et al., 2021), and missing data imputation (Luo et al., 2018).

ModernTCN is a general time series analysis model designed to enhance performance across five key time series tasks: long-term forecasting, short-term forecasting, classification, imputation, and anomaly detection. It modernizes traditional Temporal Convolutional Networks (TCNs) by enlarging the effective receptive field (ERF), drawing inspiration from computer vision advances (Liu et al., 2022a; Ding et al., 2022a) and Transformer architectures (Vaswani et al., 2017). The model also incorporates techniques from transformer-based time series models, such as patching and variable independence approaches (Nie et al., 2023).

The concept of a general time series analysis model, as seen in ModernTCN, builds on the experimental setup introduced by TimesNet (Wu et al., 2023). However, this setup has several flaws. For example, the setup often accelerates testing by splitting data into batches and discarding the last incomplete batch of the test set, which can lead to unfair comparisons and misleading results (Qiu et al., 2024). In anomaly detection, this setup employs an unusual evaluation method prior to scoring, which can lead to an overestimation of performance, as demonstrated by Kim et al. (2022). Additionally, for short-term forecasting and classification, the test set is used for validation, which compromises the integrity of the results. Alongside ModernTCN, many papers, such as GPT4TS (Zhou et al., 2023), UniTS (Gao et al., 2024), and TimeMixer++ (Wang et al., 2025), follow this general time series analysis setup.

The primary purpose of this study is to validate this setup using ModernTCN as a means. Furthermore, we extend the benchmarks and experiments to rectify the limitations of the original setup, thereby improving

the robustness and applicability of the models. Additionally, we address the oversight of global convolutional-based models (Romero et al., 2021b; Gu et al., 2021; Romero et al., 2021a; Knigge et al., 2023) in the original study. By harnessing the potential synergy between a global convolutional method (Romero et al., 2021b) and ModernTCN, we enhance the model performance on irregularly sampled data.

The contributions of this study are:

- Conducting a comprehensive evaluation of ModernTCN by comparing it against a broader set of benchmarks and through a validated and extended set of experiments, identifying and addressing issues with data handling and experimental methodology in the original setup.
- Innovating ModernTCN by integrating it with a global convolutional method, resulting in a superior model for irregularly sampled data and achieving state-of-the-art performance on the PhysioNet 2019 dataset (Reyna et al., 2019).
- Validating the claims of ModernTCN's ability to model long-range dependencies by visualizing its effective receptive field and comparing it to global convolutional methods.

2 Related Work

General time series models like TimesNet, GPT4TS, and UniTS aim to provide comprehensive solutions across various time series tasks. TimesNet introduces a unified framework for time series analysis by transforming 1D time series into 2D tensors, allowing for the modeling of complex temporal variations using 2D kernels (Wu et al., 2023). GPT4TS leverages pretrained language models, using the Frozen Pretrained Transformer (FPT) to perform various time series tasks by fine-tuning on major types of tasks (Zhou et al., 2023). UniTS employs task tokenization to integrate predictive and generative tasks into a single framework, using a modified transformer block to capture universal time series representations, enabling transferability across diverse datasets and tasks (Gao et al., 2024). All these models utilize an experimental setup that originates from TimesNet, similar to ModernTCN.

Recent studies are addressing the faulty setups in long-term forecasting and anomaly detection, which are part of the general time series analysis setup. The "Drop Last Trick," as highlighted by Qiu et al. (2024), involves discarding the last batch if it contains fewer instances than the batch size, which can lead to misleading results. Kim et al. (2022) have shown the misleading nature of point adjustment in anomaly detection. Additionally, Liu & Paparrizos (2024) proposes a comprehensive benchmark addressing the known but often ignored issues in the domain of time series anomaly detection.

Transformers have shown strong performance in time series forecasting, as demonstrated by models like PatchTST (Nie et al., 2023). However, MLP-based models have gained popularity by questioning the necessity of transformers, offering similar performance with greater efficiency, as seen in DLinear (Zeng et al., 2023). Convolution-based models, on the other hand, strike a balance by being more efficient than transformers with subquadratic time complexity and more expressive than MLPs (Luo & Wang, 2024; Wang et al., 2022).

In addition to TimesNet and ModernTCN, convolutional models like TCN (Bai et al., 2018) and MICN (Wang et al., 2022) offer distinct approaches to time series analysis. TCN, as described by Bai et al. (2018), employs causal convolutions to ensure no information leakage from future to past, allowing it to handle sequences of any length and map them to output sequences of the same length. This architecture is designed to combine simplicity with autoregressive prediction and long memory, making it effective for sequence modeling. On the other hand, MICN goes beyond causal convolution by proposing a multi-scale convolution structure that combines local features and global correlations in time series, enhancing its ability to capture complex temporal patterns.

ModernTCN is inspired by computer vision techniques to enlarge the ERF, but it overlooks a line of research involving models with global convolution capabilities, such as CKConv Romero et al. (2021b), FlexConv Romero et al. (2021a), CCNN Knigge et al. (2023), and S4 Gu et al. (2021). These models achieve substan-

tially larger ERFs through continuous kernel formulations, enabling arbitrarily large memory horizons while maintaining efficiency.

3 Methodology

3.1 ModernTCN

Figure 1: Diagram of the ModernTCN block. Adapted from Luo & Wang (2024).

ModernTCN (Luo & Wang, 2024) introduces a convolutional approach with large kernel sizes inspired by advances in computer vision (Liu et al., 2022c; Ding et al., 2022b; Liu et al., 2022b). It leverages modern convolution techniques, incorporating depthwise and pointwise convolution layers organized similarly to Transformer blocks (Vaswani et al., 2017). The depthwise convolution (DWConv) layer is responsible for learning temporal information among tokens on a per-channel basis, akin to the self-attention module in Transformers. A large kernel is used in DWConv to enhance the effective receptive field, allowing the model to capture long-term dependencies more effectively. The ConvFFN module consists of two pointwise convolution layers that adopt an inverted bottleneck structure. This module is applied twice in the architecture, first along feature dimensions and then between variables, to learn new feature representations.

The pipeline of ModernTCN is as follows:

1. **Input**: The process begins with the input time series $\mathbf{X}_{\text{in}} \in \mathbb{R}^{M \times L}$, where M is the number of variables and L is the input length.

2. **Patchify Embeddings**: The input is transformed into a higher-dimensional space by patching in a variable-independent manner, resulting in $\mathbf{X}_{\text{emb}} \in \mathbb{R}^{M \times D \times N}$, where D is the feature dimension and N is the number of patches.

3. **Backbone**: As shown in Figure 1 each ModernTCN block consists of:

   - **DWConv Layer**: Processes temporal information independently for each variable and each feature, maintaining dimensions $M \times D$.
   - **ConvFFN Module**: This module consists of two pointwise convolution layers and adopts an inverted bottleneck structure. It is applied twice:
     - **ConvFFN1**: Processes each variable independently with groups=M, learning new feature representations independently for each variable. Until this point, the entire pipeline maintains variable independence.
     - **ConvFFN2**: Processes across variables with groups=D, capturing dependencies between different variables. This is the first component in the pipeline that allows cross-variable information mixing.

   These blocks are stacked and organized in a residual manner as shown in Figure 1.

4. **Flatten and Prediction Head**:

   - **Forecasting**: The output is reshaped and passed through a prediction head to produce the final time series output $\hat{\mathbf{X}} \in \mathbb{R}^{M \times T}$, where T is the prediction length.
   - **Imputation and Anomaly Detection**: The output is reshaped to $\hat{\mathbf{X}} \in \mathbb{R}^{M \times L}$.
   - **Classification**: The representation is flattened to and passed through a projection layer with a SoftMax activation to yield the classification result $\hat{\mathbf{X}} \in \mathbb{R}^{1 \times \text{Cls}}$, where Cls is the number of classes.

5. **RevIN**: For all tasks except classification, Stationary Technique RevIN (Kim et al., 2021) is applied. It normalizes the input time series per variable with zero mean and unit standard deviation before patching and embedding. After the forward process, the mean and deviation are added back to the final prediction per variable.

This architecture leverages the decoupling of temporal, feature, and variable dimensions to improve both performance and efficiency in time series analysis, while the enlarged kernel size in DWConv enhances its ability to capture long-term dependencies.

3.2 Tasks and Datasets

Table 1: Dataset descriptions. The dataset size is organized in (train, validation, test). The column length refers to prediction length for forecasting tasks, input length for imputation, series length for classification, and sliding window length for anomaly detection. Partially adapted from Wu et al. (2023).

Tasks	Dataset	Dim(+Static)	Length	Dataset Size	Information (Frequency)
Long-term forecasting	ETTm1, ETTm2	7	{96, 192, 336, 720}	(34465, 11521, 11521)	Electricity (15 mins)
	ETTh1, ETTh2	7	{96, 192, 336, 720}	(8545, 2881, 2881)	Electricity (hourly)
	Electricity	321	{96, 192, 336, 720}	(18317, 2633, 5261)	Electricity (hourly)
	Traffic	862	{96, 192, 336, 720}	(12185, 1757, 3509)	Transportation (hourly)
	Weather	21	{96, 192, 336, 720}	(36792, 5271, 10540)	Weather (10 mins)
	Exchange	8	{96, 192, 336, 720}	(5120, 665, 1422)	Exchange rate (daily)
	ILI	7	{24, 36, 48, 60}	(617, 74, 170)	Illness (weekly)
Short-term forecasting original	M4-Yearly	1	6	(23000, 0, 23000)	Demographic
	M4-Quarterly	1	8	(24000, 0, 24000)	Finance
	M4-Monthly	1	18	(48000, 0, 48000)	Industry
	M4-Weekly	1	13	(359, 0, 359)	Macro
	M4-Daily	1	14	(4227, 0, 4227)	Micro
	M4-Hourly	1	48	(414, 0, 414)	Other
Extended	ETTm1	7	{6, 12, 24}	(34465, 11521, 11521)	Electricity (15 mins)
	ETTh1	7	{6, 12, 24}	(8545, 2881, 2881)	Electricity (hourly)
Imputation	ETTm1, ETTm2	7	96	(34465, 11521, 11521)	Electricity (15 mins)
	ETTh1, ETTh2	7	96	(8545, 2881, 2881)	Electricity (15 mins)
	Electricity	321	96	(18317, 2633, 5261)	Electricity (15 mins)
	Weather	21	96	(36792, 5271, 10540)	Weather (10 mins)
Classification original	EthanolConcentration	3	1751	(261, 0, 263)	Alcohol industry
	FaceDetection	144	62	(5890, 0, 3524)	Face (250Hz)
	Handwriting	3	152	(150, 0, 850)	Handwriting
	Heartbeat	61	405	(204, 0, 205)	Heart beat
	JapaneseVowels	12	29	(270, 0, 370)	Voice
	PEMS-SF	963	144	(267, 0, 173)	Transportation (daily)
	SelfRegulationSCP1	6	896	(268, 0, 293)	Healthcare (256Hz)
	SelfRegulationSCP2	7	1152	(200, 0, 180)	Healthcare (256Hz)
	SpokenArabicDigits	13	93	(6599, 0, 2199)	Voice (11025Hz)
	UWaveGestureLibrary	3	315	(120, 0, 320)	Gesture
Extended	Speech Commands MFCC	20	161	(24483, 5246, 5246)	Voice(16000Hz)
	Speech Commands Raw	1	16000	(24483, 5246, 5246)	Voice(16000Hz)
	PhysioNet	34(+5)	72	(28235, 6050, 6050)	Healthcare
	Character Trajectories	3	[60-182]	(1209, 213, 1436)	Handwriting
Anomaly detection	SMD	38	100	(566724, 141681, 708420)	Server machine
	MSL	55	100	(44653, 11664, 73729)	Spacecraft
	SMAP	25	100	(108146, 27037, 427617)	Spacecraft
	SWaT	51	100	(396000, 99000, 449919)	Infrastructure
	PSM	25	100	(105984, 26497, 87841)	Server machine
Extended	TSB-AD-U	1	It is curated from 23 different datasets. See Appendix A.2 for details.		
	TSB-AD-M	[2-248]	It is curated from 17 different datasets. See Appendix A.2 for details.		

Time series analysis encompasses a variety of tasks, each designed to extract different insights from sequential data. This study focuses on five primary tasks: long-term forecasting, short-term forecasting, imputation, classification, and anomaly detection. An overview of all the datasets used is given in Table 1.

Long-term forecasting involves predicting future values over an extended time horizon, often requiring the model to capture complex temporal dependencies. ModernTCN (Luo & Wang, 2024) uses datasets such as ETT (Zhou et al., 2021), Electricity (UCI), Traffic (PeM), Weather (Wet), Exchange (Lai et al., 2018), and ILI (CDC), covering real-world applications from different domains. In these multivariate forecasting tasks, the model predicts future values for all variables in the time series simultaneously.

Short-term forecasting focuses on predicting values over a shorter time frame, typically requiring the model to capture immediate trends and patterns. ModernTCN uses the M4 dataset (Makridakis, 2018), which contains univariate time series with different frequencies from different domains. We extend the experiments to ETT (Zhou et al., 2021) by adapting it for short term multivariate forecasting.

Imputation aims to fill in missing values in a time series, which is crucial for handling incomplete data and ensuring data quality. ModernTCN (Luo & Wang, 2024) uses ETT (Zhou et al., 2021), Electricity (UCI), and Weather (Wet) datasets for this task.

Classification is sequence-level that verifies the model's capacity in high-level representation learning. ModernTCN (Luo & Wang, 2024) uses 10 multivariate datasets from the UEA Time Series Classification Archive (Bagnall et al., 2018). We extend to Speech Commands (Warden, 2018) and PhysioNet (Reyna et al., 2019), following the experimental setup from CKConv (Romero et al., 2021b). Speech Commands enables us to compare ModernTCN with other convolution-based models that has global receptive fields. PhysioNet's irregular sampling and high proportion of missing values present a challenging classification task. Additionally, we extend the evaluation to variable length time series classification using Character Trajectories dataset (Bagnall et al., 2018).

Anomaly detection seeks to identify unusual patterns or outliers in a time series, which is essential for detecting abnormal events and potential issues. ModernTCN (Luo & Wang, 2024) evaluates anomaly detection performance on SMD (Su et al., 2019a), MSL (Hundman et al., 2018), SMAP (Hundman et al., 2018), SWaT (Mathur & Tippenhauer, 2016), and PSM (Abdulaal et al., 2021), covering service monitoring, space & earth exploration, and water treatment applications. We extend the evaluation to TSB-AD (Liu & Paparrizos, 2024), a comprehensive benchmark for anomaly detection consisting of 1070 high-quality time series from a diverse collection of 40 datasets (see Table 11 for details).

3.3 Experimental Setup

For the reproduction of the experiments from ModernTCN, the optimal settings specified in the paper (Luo & Wang, 2024) and the source code[1] are used. For the extended experiments we conducted a grid search over the relevant hyperparameters for each setup (See Appendix A.1 for details). For the selected hyperparameters, we reported the mean and standard deviation over 5 runs.

Upon investigating the source code, we identified issues affecting the five tasks used in Luo & Wang (2024). We fixed the errors and rerun the experiment for long-term forecasting and imputation tasks. For short-term forecasting, classification and anomaly detection, we extended the experiments to new datasets instead.

3.3.1 Issues in the Original Setup

Drop Last Trick. As pointed out by Qiu et al. (2024), many implementations of existing methods often employ a "Drop Last Trick" during the testing phase (Nie et al., 2022; Wang et al., 2022; Zhou et al., 2022; 2021), which involves discarding the last batch if it contains fewer instances than the batch size. This approach may lead to unfair comparisons by excluding part of the test data. Upon reviewing the source code, we found this trick is employed for long-term forecasting and imputation tasks. Given that most datasets in the long-term forecasting experiments use a batch size of 512, this could result in discarding a significant amount of the test set (See Appendix 14 for further analysis). Therefore, we additionally experiment without employing this trick to assess its impact on the results and ensure a fair comparison (Table 2 and Table 4).

Data Leakage from Test to Validation. As detailed in Table 1, the M4 dataset (Makridakis, 2018) for short-term forecasting and the UEA dataset (Bagnall et al., 2018) for classification are provided with only train and test splits. Upon reviewing the source code, we identify that the test set is used for validation. This practice introduces an information leakage as the model gains indirect exposure to the test data, leading to an overestimation of its generalization capabilities. Consequently, the reproduction results for these datasets are excluded from the main results section and are instead presented in Appendix E.1 and Appendix E.2.

Point Adjustment in Anomaly Detection. Reconstruction-based anomaly detection identifies anomalies when reconstruction errors between input sequences and their model-generated reconstructions exceed a threshold. The threshold is calculated by taking a specific percentile of the reconstruction error scores from the training data, ensuring that a defined percentage of the data is considered normal. But ModernTCN calculates the threshold using both training and test sets, introducing information leakage. We reproduce

[1]The official source code for ModernTCN can be found at `https://github.com/luodhhh/ModernTCN`

the experiments using thresholds calculated from training data only. Additionally, the evaluation employs "point adjustment" (PA), which considers an entire anomaly segment correctly detected if just one point within it is flagged - a protocol that Kim et al. (2022) demonstrated can severely overestimate performance. To illustrate this limitation, we implemented a naive baseline that periodically flags every 100th timepoint as anomalous, which should perform well under PA despite having no actual anomaly detection capability. 100 is set as the period in the naive approach, as the sliding window size that is reconstructed is 100.

3.3.2 Extended Setups

Extending Short-term Forecasting to ETT. Inspired by the appendices in Luo & Wang (2024), we evaluate short-term multi-variate forecasting on the ETT dataset (Zhou et al., 2021). While ModernTCN (Luo & Wang, 2024) keeps the input sequence length constant at 2x the prediction length, we explored the input sequence length as a hyperparameter, experimenting with 2x, 3x, and 4x the prediction length. We consider three prediction lengths: 6, 12, and 18 time points, selecting the best-performing input length for reporting. We did a hyperparameter search for the kernel size and employed the optimal hyperparameters from the corresponding long-term forecasting experiments for the rest. For comparison, we decided to include three top-performing models of different kinds from Qiu et al. (2024): TimesNet (convolution-based) (Wu et al., 2023), PatchTST (transformer-based) (Nie et al., 2023), and DLinear (MLP-based) (Zeng et al., 2023).

Extending Classification to Speech Commands. For the classification task, we employ the Speech Commands dataset (Warden, 2018). This dataset has two versions: Speech Commands Raw and Speech Commands MFCC. The Speech Commands Raw version consists of 105,809 one-second audio recordings of 35 spoken words sampled at 16kHz. Following the preprocessing steps of Kidger et al. (2020), we extract 34,975 recordings from ten spoken words to construct a balanced classification problem. The Speech Commands MFCC utilizes mel-frequency cepstrum coefficients extracted from the raw data, resulting in time series of length 161 and 20 channels. This dataset is frequently used by convolution-based models with global receptive fields (Romero et al., 2021b;a; Knigge et al., 2023; Gu et al., 2021).

Extending to Variable Length Time Series Classification. The CharacterTrajectories dataset is part of the UEA time series classification archive (Bagnall et al., 2018). It consists of 2,858 time series of varying lengths, each with 3 channels representing the x, y positions and the pen tip force while writing a Latin alphabet character in a single stroke. The goal is to classify which of the 20 different characters was written using the time series data. The length of the time series varies between 60 and 182. Unlike Romero et al. (2021b), we retained the original test set and created a validation split by taking 15% of the training set. This approach ensures comparability with Lee & Shin (2025). Additionally, we reran the CKConv (Romero et al., 2021b) experiments with this new setup using its official codebase[2].

Extending to Irregularly Sampled Data. The PhysioNet 2019 challenge on sepsis prediction provides an irregularly sampled, partially observed dataset with 40,335 time series of variable lengths, capturing ICU patient data (Reyna et al., 2019). Each series includes 5 static features, like age, and 34 dynamic features, such as respiration rate, with only 10.3% of values observed. Following Kidger et al. (2020); Romero et al. (2021b), we analyze the first 72 hours of patient data to predict sepsis development over their entire stay, which can last up to a month. This setup tests the model's ability to handle sparse and irregular data, creating a robust test bed for ModernTCN, which claims state-of-the-art performance on the imputation task. Similar to ModernTCN, we ablate the cross-variable component to assess its impact on handling missing values in a multivariate setting. Most importantly, we introduce an innovation in our reproducibility study by combining a global convolutional method (Knigge et al., 2023) with ModernTCN, as detailed in Table 10.

Extending Anomaly Detection to TSB-AD. Extending our anomaly detection experiments to the TSB-AD datasets and benchmark (Liu & Paparrizos, 2024) is beneficial due to its comprehensive nature, encompassing over 40 datasets and a wide array of baselines. The benchmark includes 40 time-series anomaly detection algorithms categorized into statistical, neural network-based, and foundation model-based methods. This extension addresses critical issues in the field, such as flawed datasets and biased evaluation measures, by employing VUS-PR (Volume Under the Surface of Precision-Recall) as a more reliable evaluation metric.

[2]The official CKConv codebase can be found at `https://github.com/dwromero/ckconv`

VUS-PR is threshold-independent, summarizing performance across all possible anomaly score thresholds, thus providing a holistic and stable evaluation. It is more robust to temporal misalignments and less biased towards random or noisy scores compared to traditional measures like PA-F1 and AUC-ROC. By incorporating a tolerance buffer around anomaly boundaries, VUS-PR enhances the relevance of anomaly scoring, making it a superior choice for evaluating time-series anomaly detection models.

4 Experimental Results

4.1 Results Reproducing the Original Paper

4.1.1 Long-Term Forecasting

The "Drop Last Trick" significantly impacts datasets like ETTh and Exchange, where a substantial percentage of the test set is discarded, as in Table 2. This leads to notable changes in MSE/MAE, particularly for these datasets, as a high percentage of data is dropped. Additionally, the ILI dataset exhibits significant changes in MSE/MAE, likely due to its small size (Table 1). For further analysis, please refer to the Appendix D.

All results in Table 3 are presented after the "Drop Last Trick" is corrected. PatchTST (Nie et al., 2023), TimesNet (Wu et al., 2023), and DLinear (Zeng et al., 2023) are the best-performing methods based on transformer, convolution, and MLP architectures, respectively, as identified by Qiu et al. (2024). Additionally, we include MICN (Wang et al., 2022) and TCN (Bai et al., 2018) as extra convolution-based baselines, reproduced using the codebase of Qiu et al. (2024). We also included results for two general time series analysis foundation models (Zhou et al., 2023; Gao et al., 2024) from (Li et al., 2024) for four datasets. These results are on a full-shot setting. Since these results are not available for all datasets, they are not included in the 1st/2nd place ranking count.

While PatchTST outperforms all other models based on the 1st rank count (12 times), ModernTCN achieves a very close score, frequently ranking 2nd (12 times) and occasionally 1st (6 times). When comparing ModernTCN with other convolutional-based baselines, it is evident that ModernTCN outperforms them in most cases. Lastly, foundational models generally lag behind but come close to the best (depicted in blue) on the Weather dataset.

Table 2: Long-term forecasting reproduced results of ModernTCN. Reported, Rerun and Fixed refer to original paper results, results before and after the "Drop Last Trick" is fixed respectively. Dropped % shows percentage of test data points excluded when using the trick. Δ MSE/MAE shows performance change when the trick is fixed and it is calculated as "Δ = Fixed - Rerun". Results are averaged from 4 different prediction lengths: {24, 36, 48, 60} for ILI and {96, 192, 336, 720} for others.

	ETTh1		ETTh2		ETTm1		ETTm2		Electricity		Weather		Traffic		Exchange		ILI	
Result Type	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Reported	0.404	0.420	0.322	0.379	0.351	0.380	0.253	0.314	0.156	0.253	0.224	0.264	0.396	0.270	0.302	0.366	1.440	0.786
Rerun	0.404	0.421	0.322	0.379	0.355	0.383	0.255	0.319	0.159	0.256	0.224	0.267	0.401	0.274	0.305	0.368	1.490	0.797
Fixed	0.419	0.429	0.347	0.394	0.356	0.383	0.254	0.318	0.163	0.259	0.226	0.267	0.410	0.280	0.343	0.392	2.000	0.892
Dropped %	11.10%	11.10%	11.10%	11.10%	2.23%	2.23%	2.23%	2.23%	0.25%	0.25%	1.64%	1.64%	0.60%	0.60%	14.49%	14.49%	5.88%	5.88%
Δ MSE/MAE	+0.015	+0.008	+0.025	+0.015	+0.001	0.000	-0.001	-0.001	+0.004	+0.003	+0.003	+0.001	+0.009	+0.006	+0.038	+0.024	+0.510	+0.095

Table 3: Long-term forecasting task results after the "Drop Last Trick" is fixed. All results averaged from 4 different prediction lengths: {24, 36, 48, 60} for ILI and {96, 192, 336, 720} for others. Lower MSE/MAE indicates better performance. Bold indicates best performance, and underline indicates second-best performance. "-" in the 1st/2nd column means not applicable for ranking, and "-" in MSE/MAE columns indicates no results reported in Li et al. (2024).

		ETTh1		ETTh2		ETTm1		ETTm2		Electricity		Weather		Traffic		Exchange		ILI	
Model	**1st/2nd**	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ModernTCN	6/10	0.419	0.429	**0.347**	0.394	0.356	0.383	**0.254**	0.318	**0.163**	**0.259**	0.226	0.267	0.410	**0.280**	0.343	**0.392**	2.000	0.892
PatchTST	12/4	**0.411**	**0.428**	**0.347**	**0.389**	**0.349**	0.381	0.255	**0.313**	**0.163**	0.261	**0.225**	**0.262**	**0.405**	0.283	0.352	0.397	**1.770**	**0.859**
TimesNet	0/1	0.459	0.455	0.394	0.416	0.430	0.428	0.294	0.332	0.186	0.287	0.261	0.287	0.626	0.328	0.421	0.442	2.174	0.951
DLinear	1/2	0.420	0.432	0.492	0.478	0.354	**0.377**	0.259	0.324	0.167	0.264	0.239	0.290	0.434	0.295	0.349	0.411	2.185	1.040
MICN	1/1	0.420	0.447	0.482	0.471	0.355	0.383	0.294	0.357	0.179	0.290	0.239	0.289	0.539	0.313	**0.320**	0.396	2.368	1.049
TCN	0/0	1.004	0.787	3.398	1.545	1.190	0.840	2.465	1.177	0.385	0.448	0.409	0.422	1.061	0.556	1.785	1.093	4.320	1.359
UniTS	-	0.453	0.461	-	-	-	-	-	-	-	-	0.224	0.268	-	-	0.609	0.535	2.845	1.196
GPT4TS	-	0.420	0.436	-	-	-	-	-	-	-	-	0.231	0.267	-	-	0.475	0.471	3.764	1.412

4.1.2 Imputation

The results presented in Table 4 demonstrate that due to the smaller batch size used in the imputation task, as referenced in Table 14, the percentage of dropped points from the test set is minimal. This minimal drop percentage explains why the results remained unchanged after the "Drop Last Trick" was corrected.

In our comparison of ModernTCN with TimesNet, PatchTST, DLinear, and MICN, ModernTCN clearly outperforms all other methods (Table 5). TimesNet ranks second, which aligns with the fact that both ModernTCN and TimesNet are designed for general time series analysis, whereas PatchTST, DLinear, and MICN are specifically developed for time series forecasting. Additionally, PatchTST and DLinear are variable-independent models, while this task is inherently multivariate. The ability to model cross-variable interactions may facilitate the imputation of missing values by leveraging information from existing variables at the same time point, thereby modeling inter-variate dependencies effectively.

Table 4: Imputation task reproduced results of ModernTCN. Reported, Rerun and Fixed refer to original paper results, results before and after the "Drop Last Trick" is fixed respectively. Dropped % shows percentage of test data points excluded when using the trick. Δ MSE/MAE shows performance change when the trick is fixed and it is calculated as "Δ = Fixed - Rerun". Results are averaged from 4 different mask ratios {12.5%, 25%, 37.5%, 50%}.

	ETTh1		ETTh2		ETTm1		ETTm2		Electricity		Weather	
Result Type	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Reported	0.050	0.150	0.042	0.131	0.020	0.093	0.019	0.082	0.073	0.187	0.027	0.044
Rerun	0.050	0.150	0.042	0.131	0.021	0.094	0.020	0.083	0.073	0.186	0.027	0.044
Fixed	0.050	0.150	0.042	0.131	0.021	0.094	0.020	0.083	0.073	0.186	0.027	0.044
Dropped %	0.03%	0.03%	0.03%	0.03%	0.01%	0.01%	0.01%	0.01%	0.25%	0.25%	0.11%	0.11%
Δ MSE/MAE	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000

Table 5: Imputation task results after the "Drop Last Trick" is fixed. We randomly mask {12.5%, 25%, 37.5%, 50%} time points in length-96 time series. The results are averaged from 4 different mask ratios. A lower MSE or MAE indicates better performance. Bold indicates best performance, and underline indicates second-best performance.

	ETTh1		ETTh2		ETTm1		ETTm2		Electricity		Weather	
Model	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ModernTCN	**0.050**	**0.150**	**0.042**	**0.131**	**0.021**	**0.094**	**0.020**	**0.083**	**0.073**	**0.186**	**0.027**	**0.044**
TimesNet	0.089	0.199	0.051	0.149	0.027	0.107	0.022	0.090	0.094	0.211	0.031	0.056
PatchTST	0.134	0.238	0.066	0.164	0.045	0.133	0.028	0.098	0.091	0.209	0.033	0.057
DLinear	0.201	0.306	0.142	0.259	0.093	0.206	0.096	0.208	0.132	0.260	0.052	0.110
MICN	0.125	0.250	0.205	0.307	0.070	0.182	0.144	0.249	0.119	0.247	0.056	0.128

4.1.3 Anomaly Detection

As shown in Table 6, the rerun results generally align with those reported in the original paper. When eliminating data leakage by calculating thresholds using only training data, we observe a slight performance degradation. Most notably, our naive baseline—which simply flags every 100th timepoint as anomalous—outperforms ModernTCN across most datasets, achieving the highest average F1 score. This finding raises significant questions about the effectiveness of sophisticated models for anomaly detection when evaluated using point adjustment protocols, supporting the critique by Kim et al. (2022) that such evaluation methods can severely overestimate model performance.

4.2 Results Beyond the Original Paper

4.2.1 Short-Term Forecasting on ETT

Referring to the results presented in Table 7, ModernTCN exhibits competitive performance, particularly in the ETTm1 dataset, where it achieves the best results for very short-term forecasting. This aligns with

Table 6: Anomaly Detection task. P: precision, R: recall, F1: F1-score (all values in %). Reported: scores from Luo & Wang (2024). Rerun: naively reproduced results. No Leakage: results calculating the thresholds using only training data. Naive: baseline approach. Bold: best score between valid models (No Leakage and Naive). Avg.: average F1 score across all 5 datasets.

Model	SMD			MSL			SMAP			SWaT			PSM			Avg.
	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1	
Reported	87.86	83.85	85.81	83.94	85.93	84.92	93.17	57.69	71.26	91.83	95.98	93.86	98.09	96.38	97.23	86.62
Rerun	87.43	81.64	84.44	89.59	74.94	81.61	90.82	55.93	69.23	95.77	90.28	92.94	98.65	94.57	96.57	84.96
No Leakage	78.42	83.76	81.00	86.07	77.12	81.35	92.31	55.42	69.26	**93.45**	93.16	93.30	**98.62**	**94.58**	**96.56**	84.29
Naive	**79.86**	**91.40**	**85.24**	**90.14**	**80.76**	**85.19**	**93.27**	**94.48**	**93.87**	93.05	**96.93**	**94.95**	97.32	94.55	95.91	**91.03**

findings from Yang et al. (2024), which demonstrated ModernTCN's effectiveness for ultra-short-term multi-step prediction tasks. However, in ETTh1 dataset, PatchTST demonstrates superior performance.

Table 7: Short-Term Forecasting Results on ETT (ETTh1 and ETTm1). 6x1h, 12x1h, and 18x1h refer to steps ahead forecasting for ETTh1, while 6x15m, 12x15m, and 18x15m refer to steps ahead forecasting for ETTm1. Results are averaged over 5 runs, and both mean and standard deviation are reported.

Model	6x15m		12x15m		18x15m		6x1h		12x1h		18x1h	
	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ModernTCN	**0.127** $\pm$ 0.001	**0.216** $\pm$ 0.001	**0.204** $\pm$ 0.001	**0.269** $\pm$ 0.001	**0.220** $\pm$ 0.001	**0.296** $\pm$ 0.001	0.300 $\pm$ 0.006	0.350 $\pm$ 0.003	0.300 $\pm$ 0.001	0.346 $\pm$ 0.001	0.308 $\pm$ 0.001	0.351 $\pm$ 0.001
PatchTST	0.138 $\pm$ 0.003	0.222 $\pm$ 0.001	0.244 $\pm$ 0.004	0.295 $\pm$ 0.005	0.240 $\pm$ 0.002	0.312 $\pm$ 0.001	**0.253** $\pm$ 0.003	**0.321** $\pm$ 0.001	**0.278** $\pm$ 0.003	**0.333** $\pm$ 0.002	**0.288** $\pm$ 0.002	**0.344** $\pm$ 0.002
TimesNet	0.144 $\pm$ 0.002	0.235 $\pm$ 0.001	0.231 $\pm$ 0.003	0.295 $\pm$ 0.003	0.256 $\pm$ 0.003	0.326 $\pm$ 0.001	0.316 $\pm$ 0.002	0.367 $\pm$ 0.001	0.337 $\pm$ 0.007	0.383 $\pm$ 0.004	0.348 $\pm$ 0.006	0.393 $\pm$ 0.004
DLinear	0.146 $\pm$ 0.001	0.233 $\pm$ 0.001	0.287 $\pm$ 0.001	0.323 $\pm$ 0.001	0.303 $\pm$ 0.001	0.347 $\pm$ 0.001	0.284 $\pm$ 0.002	0.339 $\pm$ 0.001	0.292 $\pm$ 0.001	0.341 $\pm$ 0.002	0.300 $\pm$ 0.001	0.347 $\pm$ 0.001

4.2.2 Classification on Speech Commands

The results indicate that while ModernTCN's performance has improved, it generally lags behind global convolutional models, as shown in Table 8. ModernTCN exhibits less parameter efficiency due to its design, which maintains an extra dimension for each variable throughout the pipeline. This results in the model dimension being the product of the selected model dimension and the number of variables. In the case of Speech Commands MFCC, with 20 variables and a feature dimension of 32, the model dimension is 20 x 32 = 640. This contrasts with CCNN (Knigge et al., 2023), which uses depthwise separable convolutions but projects all variables to the model dimension at the beginning, resulting in greater parameter efficiency.

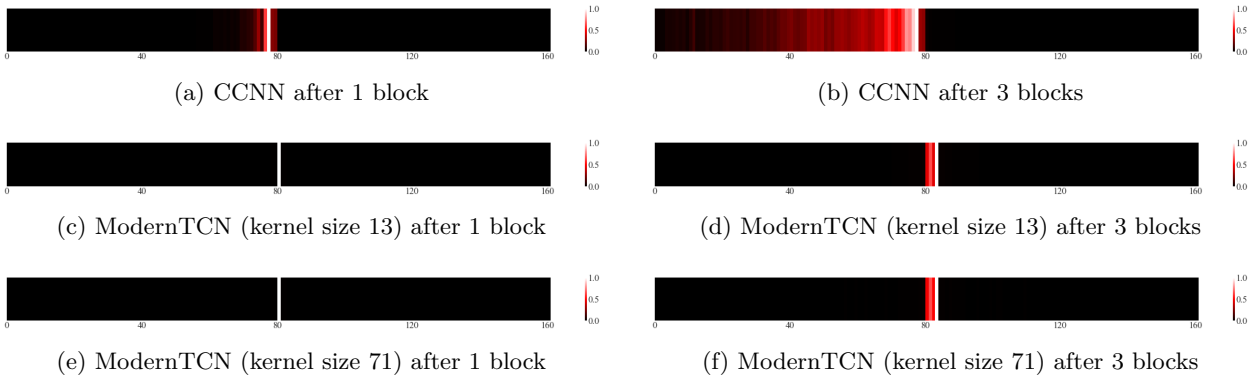
For the Speech Commands Raw dataset, the model size selected for ModernTCN is 512. However, with only one variable, the model has only 500k parameters. This setup is particularly challenging due to the sequence length of 16000, which is very long. In this context, ModernTCN performs better than CKConv, thanks to downsampling the raw signal using a high stride during embedding, along with an increased model dimension of 512, which enhances the model's expressivity. Despite these improvements, ModernTCN still falls short of the more advanced global convolutional models.

Following the ERF visualization methodology from Kim et al. (2023) and Ding et al. (2022a) as adapted in Luo & Wang (2024), we sample 50 sequences from the Speech Commands MFCC validation set to visualize the relative gradient contribution of each input timepoint to the center of the feature map after applying a certain number of model blocks. The visualization in Figure 2 shows these contributions for input sequence length 161, with lighter areas indicating stronger influence.

CCNN{4,140} (Knigge et al., 2023), being a global convolutional model with a global kernel, is better at modeling long-range dependencies and achieves better parameter efficiency through its global kernel design. As a result, it achieves a significantly larger ERF with fewer parameters (200K vs. 4M). This advantage is reflected in the classification performance in Table 8. See Appendix B for detailed methodology and further analysis.

4.2.3 Variable Length Time Series Classification

Both ModernTCN and CKConv demonstrate strong performance in the variable length time series classification task (Table 9), outperforming the best baselines from Lee & Shin (2025). The high scores achieved by these models can be attributed to the nature of the task. In this context, convolutional models like Mod-



(a) CCNN after 1 block

(b) CCNN after 3 blocks

(c) ModernTCN (kernel size 13) after 1 block

(d) ModernTCN (kernel size 13) after 3 blocks

(e) ModernTCN (kernel size 71) after 1 block

(f) ModernTCN (kernel size 71) after 3 blocks

Figure 2: Comparison of effective receptive fields (ERF) for different models. Each row shows a different model configuration, while columns show the ERF after 1 block (left) and 3 blocks (right).

Table 8: Classification Results on Speech Commands - Accuracy

Model	Size	Speech Commands MFCC	Speech Commands Raw
CKCNN-Seq (Romero et al., 2021b)	98K	0.953	0.717
S4 (Gu et al., 2021)	300K	0.940	0.983
FlexTCN-6 (Romero et al., 2021a)	375K	0.977	0.917
CCNN{4,140} (Knigge et al., 2023)	200K	0.950	0.983
CCNN{6,380} (Knigge et al., 2023)	2M	**0.980**	**0.984**
ModernTCN	4M/500k	0.922 ± 0.001	0.843 ± 0.007

ernTCN and CKConv benefit from their ability to extract features through template matching as they slide over the time dimension. CKConv, in particular, excels in this task, likely due to the continuous nature of the handwriting trajectories it models. By representing the 1D convolutional kernel as a continuous function, CKConv effectively captures the underlying patterns in the data, leading to its exceptional performance.

Table 9: Results of Variable Length Time Series Classification (Accuracy and MAP) over 5 runs, with values representing the mean and standard deviation. ModernTCN and CKConv were adapted, while the other baselines were selected from (Lee & Shin, 2025) as the best variants/models for both metrics.

Model	Accuracy (%)	MAP (%)
ModernTCN	95.02 ± 0.9	94.42 ± 0.9
CKConv	99.07 ± 0.3	99.00 ± 0.3
Refined TIP	78.11 ± 1.18	83.35 ± 0.83
InceptionTime (IP)	82.81 ± 0.40	69.28 ± 0.45

4.2.4 Irregularly Sampled Time Series Classification

Sepsis prediction is a critical task, as early detection significantly improves patient outcomes. The PhysioNet Sepsis Prediction dataset (Reyna et al., 2019) presents a challenging test bed due to its irregularly sampled nature, which results in missing values. CKConv demonstrates strong performance on this task, surpassing models specifically designed for irregularly sampled data (Kidger et al., 2020).

The ability of ModernTCN to handle missing values is evidenced by its superior performance in the imputation tasks within this study. This capability extends to irregularly sampled data, as shown in Table 10. Additionally, ModernTCN's proficiency in mixing information between variables through its cross-variable component (ConvFFN2) is crucial, as its removal leads to a performance drop. This is indicated as "ModernTCN - Variable Independent," where the model becomes variable-independent until the fully connected projection layer at the end.

Both ModernTCN and CKConv exhibit strong performance on data with missing values. To harness their synergistic potential, we combined them by employing CKConv to embed the raw signals within ModernTCN, while retaining the ModernTCN pipeline for subsequent processing. This approach, denoted as "CKConv + ModernTCN," outperforms all other baselines. This combination leverages two key properties of CKConv:

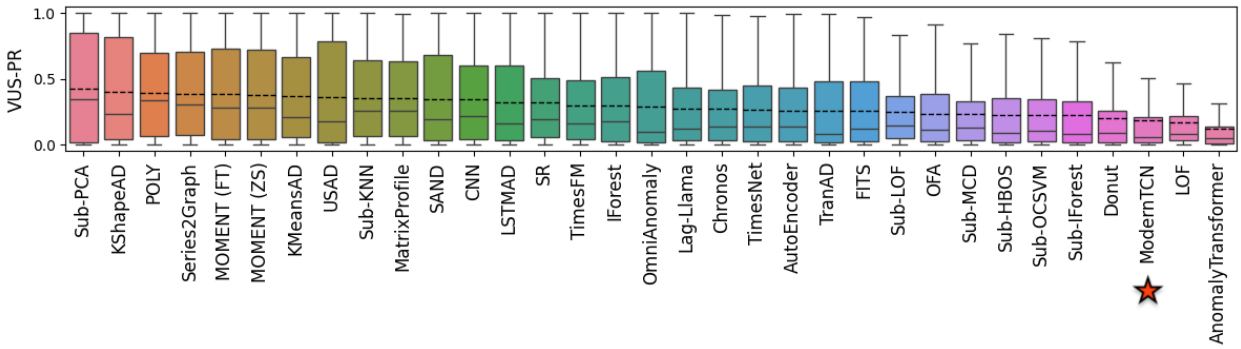
(1) the continuous nature of the kernel, which inherently handles irregularly sampled data, and (2) parameter efficiency, as the parameter size of the continuous kernel is decoupled from the input sequence length. This innovative combination could serve as a promising starting point for future studies, particularly in testing on irregularly sampled datasets, including those with very long sequence lengths.

Table 10: Comparison of Model Performance on the PhysioNet Dataset (AUC ± Std Dev)

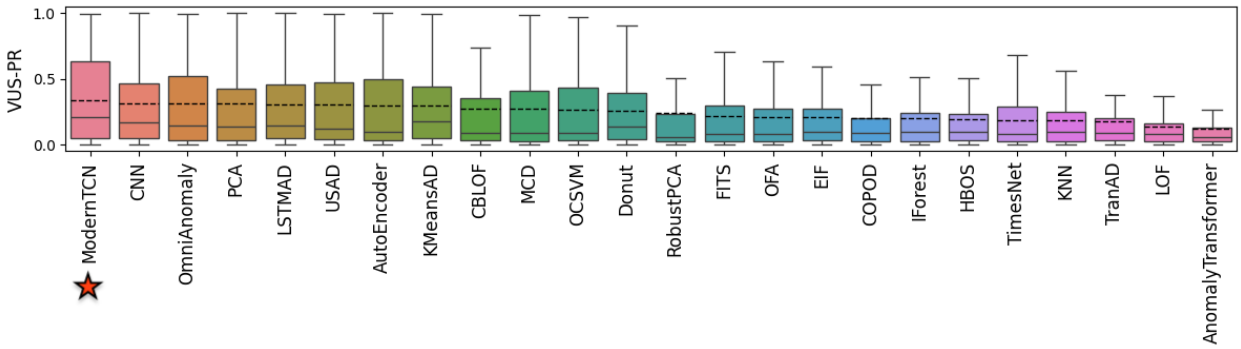
Model	**AUC ± Std Dev**
CKConv	0.883 ± 0.011
ModernTCN	0.893 ± 0.002
ModernTCN - Variable Independent	0.872 ± 0.003
CKConv + ModernTCN	0.904 ± 0.004

4.2.5 Anomaly Detection

In the context of anomaly detection, ModernTCN's performance varies significantly between univariate and multivariate time series datasets. On the univariate TSB-AD-U dataset (Figure 3(a)), ModernTCN ranks among the lowest-performing methods, placing third to last. This result aligns with the findings of Liu & Paparrizos (2024), which suggest that statistical methods often provide more effective and robust solutions for univariate time-series anomaly detection. Conversely, on the multivariate TSB-AD-M dataset (Figure 3(b)), ModernTCN emerges as the best-performing method. This success can be attributed to the extensive engineering and design considerations within ModernTCN that specifically address the complexities of multivariate time series data. These findings highlight the model's strengths in handling multivariate scenarios while also underscoring the challenges it faces in univariate contexts.



(a) Accuracy evaluation on univariate TSB-AD-U.



(b) Accuracy evaluation on multivariate TSB-AD-M.

Figure 3: Accuracy evaluation on univariate and multivariate time series anomaly detection. In the boxplot, the mean value is marked by a dashed line and the median by a solid line. Adapted from (Liu & Paparrizos, 2024).

5 Discussion

This study revisited and extended the evaluation of ModernTCN (Luo & Wang, 2024), providing a comprehensive analysis of its performance across various time series tasks. While ModernTCN demonstrates competitive results, its claim of consistent state-of-the-art performance is nuanced by its sensitivity to experimental setups and data handling. By addressing methodological issues such as the "Drop Last Trick" in long-term forecasting, we found that ModernTCN is outperformed by models like PatchTST in several scenarios. Our anomaly detection experiments further revealed that ModernTCN lags behind statistical methods for univariate data but excels in multivariate contexts, highlighting its design strengths.

The study also introduced an architectural improvement by integrating ModernTCN with a global continuous kernel convolutional method, achieving state-of-the-art performance on the PhysioNet 2019 dataset. This innovative combination opens new avenues for future research, particularly in exploring its application to other irregularly sampled datasets and those with extended sequence lengths, potentially enhancing the model's adaptability and performance in diverse scenarios.

Our analysis of effective receptive fields (ERFs) showed that while ModernTCN claims to enhance receptive fields, it achieves smaller ERFs compared to continuous kernel approaches like CCNN (Knigge et al., 2023). Additionally, ModernTCN requires significantly more parameters due to its design of maintaining an extra variable dimension alongside the time and feature dimensions, which could make the model too heavy and unsuitable for tasks with a large number of variables.

The cross-variable component (ConvFFN2) was validated as crucial for handling missing data, as demonstrated by our ablation study on PhysioNet. This aligns with the findings of the original ModernTCN study ablating this component on the imputation task.

5.1 What was easy?

The extensive experiments and detailed reporting in the ModernTCN paper made reproduction straightforward. Most hyperparameters were clearly specified for each task, and the great majority of the code was well-written, clear, and well-documented, facilitating a smooth execution of the experiments.

5.2 What was difficult?

The missing ERF visualization code presented a challenge. Additionally, the experimental setup for anomaly detection was not thoroughly detailed in the original paper. The work by Kim et al. (2022) was instrumental in helping us understand and address the issues in that task. The extensive experimental scope, coupled with the additional experiments conducted in this study, also presented a significant time investment, requiring substantial computational resources and meticulous attention to detail.

5.3 Contact with the authors

The authors were contacted via email to request the code for ERF visualization and to discuss the flaws in the experimental setup. Unfortunately, no response was received.

References

Illness. https://gis.cdc.gov/grasp/fluview/fluportaldashboard.html.

Iops. http://iops.ai/dataset_detail/?id=10.

Traffic. http://pems.dot.ca.gov/.

Tao. https://www.pmel.noaa.gov/.

Electricity. https://archive.ics.uci.edu/ml/datasets/ElectricityLoadDiagrams20112014.

Weather. https://www.bgc-jena.mpg.de/wetter/.

Ahmed Abdulaal, Zhuanghua Liu, and Tomer Lancewicki. Practical approach to asynchronous multivariate time series anomaly detection and localization. *KDD*, 2021.

S. Ahmad, A. Lavin, S. Purdy, and Z. Agha. Unsupervised real-time anomaly detection for streaming data. *Neurocomputing*, 262:134–147, 2017.

M. Bachlin, M. Plotnik, D. Roggen, I. Maidan, J. M. Hausdorff, N. Giladi, and G. Troster. Wearable assistant for parkinson's disease patients with the freezing of gait symptom. *IEEE Transactions on Information Technology in Biomedicine*, 14(2):436–446, 2009.

Anthony Bagnall, Jason Lines, Aaron Bostrom, James Large, and Eamonn Keogh. The uea multivariate time series classification archive. *arXiv preprint arXiv:1811.00075*, 2018.

Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *CoRR*, abs/1803.01271, 2018. URL http://arxiv.org/abs/1803.01271.

Kaifeng Bi, Lingxi Xie, Hengheng Zhang, Xin Chen, Xiaotao Gu, and Qi Tian. Accurate medium-range global weather forecasting with 3d neural networks. *Nature*, 619(7970):533–538, 2023.

P. Boniol, J. Paparrizos, T. Palpanas, and M. J. Franklin. Sand: streaming subsequence anomaly detection. *Proceedings of the VLDB Endowment*, 14(10):1717–1729, 2021.

Xiaohan Ding, Xiangyu Zhang, Jungong Han, and Guiguang Ding. Scaling up your kernels to 31x31: Revisiting large kernel design in cnns. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11963–11975, 2022a.

Xiaohan Ding, Xiangyu Zhang, Jungong Han, and Guiguang Ding. Scaling up your kernels to 31x31: Revisiting large kernel design in cnns. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11963–11975, 2022b.

P. Filonov, A. Lavrentyev, and A. Vorontsov. Multivariate industrial time series with cyber-attack simulation: Fault detection using an lstm-based predictive data model. *arXiv preprint arXiv:1612.06676*, 2016.

P. Fleith. Controlled anomalies time series (cats) dataset, Sep 2023. Accessed: 2023-09-01.

Shanghua Gao, Teddy Koker, Owen Queen, Thomas Hartvigsen, Theodoros Tsiligkaridis, and Marinka Zitnik. Units: Building a unified time series model. *Advances in Neural Information Processing Systems*, 2024.

A. L. Goldberger, L. A. Amaral, L. Glass, J. M. Hausdorff, P. C. Ivanov, R. G. Mark, J. E. Mietus, G. B. Moody, C.-K. Peng, and H. E. Stanley. Physiobank, physiotoolkit, and physionet: components of a new research resource for complex physiologic signals. *circulation*, 101(23):e215–e220, 2000.

S. D. Greenwald. Improved detection and classification of arrhythmias in noise-corrupted electrocardiograms using contextual information. 1990. Thesis, Massachusetts Institute of Technology, Accepted: 2005-10-07T20:45:22Z.

Albert Gu, Karan Goel, and Christopher Ré. Efficiently modeling long sequences with structured state spaces. *arXiv preprint arXiv:2111.00396*, 2021.

Kyle Hundman, Valentino Constantinou, Christopher Laporte, Ian Colwell, and Tom Soderström. Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding. *KDD*, 2018.

V. Jacob, F. Song, A. Stiegler, B. Rad, Y. Diao, and N. Tatbul. Exathlon: A benchmark for explainable anomaly detection over time series. *Proceedings of the VLDB Endowment*, 14(11):2613–2626, 2021.

E. Keogh, J. Lin, S.-H. Lee, and H. V. Herle. Finding the most unusual time series subsequence: algorithms and applications. *Knowledge and Information Systems*, 11:1–27, 2007.

Patrick Kidger, James Morrill, James Foster, and Terry Lyons. Neural controlled differential equations for irregular time series. *arXiv preprint arXiv:2005.08926*, 2020.

Bum Jun Kim, Hyeyeon Choi, Hyeonah Jang, Dong Gu Lee, Wonseok Jeong, and Sang Woo Kim. Dead pixel test using effective receptive field. *Pattern Recognition Letters*, 167:149–156, 2023.

Siwon Kim, Kukjin Choi, Hyun-Soo Choi, Byunghan Lee, and Sungroh Yoon. Towards a rigorous evaluation of time-series anomaly detection. In *Proceedings of the 36th AAAI Conference on Artificial Intelligence (AAAI)*, pp. 7194–7201, 2022.

Taesung Kim, Jinhee Kim, Yunwon Tae, Cheonbok Park, Jang-Ho Choi, and Jaegul Choo. Reversible instance normalization for accurate time-series forecasting against distribution shift. In *International Conference on Learning Representations*, 2021.

Dani Kiyasseh, Tingting Zhu, and David A. Clifton. Clocs: Contrastive learning of cardiac signals across space, time, and patients. In *International Conference on Machine Learning*, pp. 5606–5615. PMLR, 2021.

D. M. Knigge, D. W. Romero, A. Gu, E. Gavves, E. J. Bekkers, J. M. Tomczak, and J. J. Sonke. Modelling long range dependencies in nd: From task-specific to a general purpose cnn. *arXiv preprint arXiv:2301.10540*, 2023.

Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. Modeling long-and short-term temporal patterns with deep neural networks. In *SIGIR*, 2018.

K.-H. Lai, D. Zha, J. Xu, Y. Zhao, G. Wang, and X. Hu. Revisiting time series outlier detection: Definitions and benchmarks. In *Thirty-fifth conference on neural information processing systems datasets and benchmarks track (round 1)*, 2021.

N. Laptev, S. Amizadeh, and Y. Billawala. S5 - a labeled anomaly detection dataset, version 1.0(16m), Mar 2015.

Hyeonsu Lee and Dongmin Shin. Beyond information distortion: Imaging variable-length time series data for classification. *Sensors*, 25(3):621, 2025. doi: 10.3390/s25030621. URL `https://doi.org/10.3390/s25030621`. This article belongs to the Topic AI and Data-Driven Advancements in Industry 4.0.

Zhe Li, Xiangfei Qiu, Peng Chen, Yihang Wang, Hanyin Cheng, Yang Shu, Jilin Hu, Chenjuan Guo, Aoying Zhou, Qingsong Wen, Christian S. Jensen, and Bin Yang. Foundts: Comprehensive and unified benchmarking of foundation models for time series forecasting. 2024.

Qinghua Liu and John Paparrizos. The elephant in the room: Towards a reliable time-series anomaly detection benchmark. In *NeurIPS 2024 D&B*, The Ohio State University, 2024.

Shiwei Liu, Tianlong Chen, Xiaohan Chen, Xuxi Chen, Qiao Xiao, Boqian Wu, Mykola Pechenizkiy, Decebal Mocanu, and Zhangyang Wang. More convnets in the 2020s: Scaling up kernels beyond 51x51 using sparsity. *arXiv preprint arXiv:2207.03620*, 2022a.

Shiwei Liu, Tianlong Chen, Xiaohan Chen, Xuxi Chen, Qiao Xiao, Boqian Wu, Mykola Pechenizkiy, Decebal Mocanu, and Zhangyang Wang. More convnets in the 2020s: Scaling up kernels beyond 51x51 using sparsity. *arXiv preprint arXiv:2207.03620*, 2022b.

Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11976–11986, 2022c.

Donghao Luo and Xue Wang. Moderntcn: A modern pure convolution structure for general time series analysis. *International Conference on Learning Representations (ICLR)*, 2024.

Yonghong Luo, Xiangrui Cai, Ying Zhang, Jun Xu, and Yuan Xiaojie. Multivariate time series imputation with generative adversarial networks. In *Advances in Neural Information Processing Systems 31 (NeurIPS)*, 2018.

Spyros Makridakis. M4 dataset. `https://github.com/M4Competition/M4-methods/tree/master/Dataset`, 2018.

Aditya P. Mathur and Nils Ole Tippenhauer. Swat: a water treatment testbed for research and training on ics security. In *CySWATER*, 2016.

S. Moritz, F. Rehbach, S. Chandrasekaran, M. Rebolledo, and T. Bartz-Beielstein. Gecco industrial challenge 2018 dataset: A water quality dataset for the 'internet of things: Online anomaly detection for drinking water quality'competition at the genetic and evolutionary computation conference 2018, kyoto, japan, 2018. Kyoto, Japan.

Y. Nie, N. Nguyen, P. Sinthong, and J. Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. *International Conference on Learning Representations (ICLR)*, 2023.

Yuqi Nie, Nam H Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. *arXiv preprint arXiv:2211.14730*, 2022.

Boris N Oreshkin, Dmitri Carpov, Nicolas Chapados, and Yoshua Bengio. N-beats: Neural basis expansion analysis for interpretable time series forecasting. In *International Conference on Learning Representations*, 2020.

Xiangfei Qiu, Jilin Hu, Lekui Zhou, Xingjian Wu, Junyang Du, Buang Zhang, Chenjuan Guo, Aoying Zhou, Christian S. Jensen, Zhenli Sheng, and Bin Yang. Tfb: Towards comprehensive and fair benchmarking of time series forecasting methods. 2024.

Matthew A Reyna, Chris Josef, Salman Seyedi, Russell Jeter, Supreeth P Shashikumar, M Brandon Westover, Ashish Sharma, Shamim Nemati, and Gari D Clifford. Early prediction of sepsis from clinical data: the physionet/computing in cardiology challenge 2019. In *2019 Computing in Cardiology (CinC)*, 2019.

D. Roggen, A. Calatroni, M. Rossi, T. Holleczek, K. Förster, G. Tröster, P. Lukowicz, D. Bannach, G. Pirkl, and A. Ferscha. Collecting complex activity datasets in highly rich networked sensor environments. In *2010 Seventh international conference on networked sensing systems (INSS)*, pp. 233–240. IEEE, 2010.

David W. Romero, R. J. Bruintjes, J. M. Tomczak, E. J. Bekkers, M. Hoogendoorn, and J. C. van Gemert. Flexconv: Continuous kernel convolutions with differentiable kernel sizes. *arXiv preprint arXiv:2110.08059*, 2021a.

David W Romero, Anna Kuzina, Elias J Bekkers, J.M. Tomczak, and M. Hoogendoorn. Ckconv: Continuous kernel convolution for sequential data. *arXiv preprint arXiv:2102.02611*, 2021b.

I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani. Toward generating a new intrusion detection dataset and intrusion traffic characterization. In *ICISSp*, volume 1, pp. 108–116, 2018.

H. Si, C. Pei, H. Cui, J. Yang, Y. Sun, S. Zhang, J. Li, H. Zhang, J. Han, and D. Pei. Timeseriesbench: An industrial-grade benchmark for time series anomaly detection models. *arXiv preprint arXiv:2402.10802*, 2024.

Ya Su, Y. Zhao, Chenhao Niu, Rong Liu, W. Sun, and Dan Pei. Robust anomaly detection for multivariate time series through stochastic recurrent neural network. *KDD*, 2019a.

Ya Su, Youjian Zhao, Chenhao Niu, Rong Liu, Wei Sun, and Dan Pei. Robust anomaly detection for multivariate time series through stochastic recurrent neural network. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery Data Mining (KDD)*, pp. 2828–2837, 2019b. doi: 10.1145/3292500.3330672.

M. Thill, W. Konen, and T. Bäck. Markusthill/mgab: The mackey-glass anomaly benchmark, Apr 2020.

L. Tran, L. Fan, and C. Shahabi. Distance-based outlier detection in data streams. *Proceedings of the VLDB Endowment*, 9(12):1089–1100, 2016.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pp. 5998–6008, 2017.

A. von Birgelen and O. Niggemann. Anomaly detection and localization for cyber-physical production systems with self-organizing maps. *IMPROVE-Innovative Modelling Approaches for Production Systems to Raise Validatable Efficiency: Intelligent Methods for the Factory of the Future*, pp. 55–71, 2018.

Huiqiang Wang, Jian Peng, Feihu Huang, Jince Wang, Junhui Chen, and Yifei Xiao. Micn: Multi-scale local and global context modeling for long-term series forecasting. In *ICLR*, 2022.

Shiyu Wang, Jiawei Li, Xiaoming Shi, Zhou Ye, Baichuan Mo, Wenze Lin, Ju Shengtong, Zhixuan Chu, and Ming Jin. Timemixer++: A general time series pattern machine for universal predictive analysis. *International Conference on Learning Representations (ICLR)*, 2025.

Pete Warden. Speech commands: A dataset for limited-vocabulary speech recognition. *arXiv preprint arXiv:1804.03209*, 2018.

Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. Timesnet: Temporal 2d-variation modeling for general time series analysis. *arXiv preprint arXiv:2210.02186*, 2023.

R. Wu and E. J. Keogh. Current time series anomaly detection benchmarks are flawed and are creating the illusion of progress. *IEEE transactions on knowledge and data engineering*, 35(3):2421–2429, 2021.

Mao Yang, Xiangyu Li, Zifen Han, Yong Sun, and Xiqiang Chang. Ultra-short-term multi-step prediction of wind power based on moderntcn and multi-task learning. In *The 9th International Conference on Power and Renewable Energy*, 2024.

Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? *AAAI*, 2023.

S. Zhang, Z. Zhong, D. Li, Q. Fan, Y. Sun, M. Zhu, Y. Zhang, D. Pei, J. Sun, and Y. Liu. Efficient kpi anomaly detection through transfer learning for large-scale web services. *IEEE Journal on Selected Areas in Communications*, 40(8):2440–2455, 2022.

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. *AAAI*, 2021.

Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting. *International Conference on Machine Learning (ICML)*, 2022.

Tian Zhou, Peisong Niu, Liang Sun, and Rong Jin. One fits all: Power general time series analysis by pretrained lm. *Advances in Neural Information Processing Systems*, 2023.

Appendix

Contents

1 Introduction **1**

2 Related Work **2**

3 Methodology **3**

3.1 ModernTCN . . . 3

3.2 Tasks and Datasets . . . 4

3.3 Experimental Setup . . . 5

4 Experimental Results **7**

4.1 Results Reproducing the Original Paper . . . 7

4.2 Results Beyond the Original Paper . . . 8

5 Discussion **12**

5.1 What was easy? . . . 12

5.2 What was difficult? . . . 12

5.3 Contact with the authors . . . 12

Appendix **17**

A Experimental details **18**

A.1 Hyperparameters on the Extended Setups . . . 18

A.2 Anomaly Detection Datasets of TSB-U-AD and TSB-M-AD . . . 19

B Visualizing the ERF **19**

C Full Results **20**

C.1 Long-Term Forecasting . . . 20

C.2 Imputation . . . 20

D Drop Last Trick **21**

E Excluded Results **21**

E.1 M4 Dataset Results . . . 22

E.2 UEA Dataset Results . . . 22

F Short-Term Forecasting on ETT **22**

A Experimental details

A.1 Hyperparameters on the Extended Setups

In each setup, we applied a grid search over the combination of considered parameters. Evaluations were conducted on the validation set and run for 10 epochs. The selection of hyperparameters was based on the best validation score achieved within these 10 epochs.

Speech Commands (MFCC): The parameters included patch size and stride options of 1,1; 8,4; and 16,8, with the selected configuration being 1,1. The dimensions considered were 32, 64, and 128, with 32 being chosen. Kernel sizes of 13, 31, 51, and 71 were evaluated, with 13 being optimal. For learning rate scheduling, we explored exponential decay (type1), step decay (type2), and delayed exponential decay (type3), selecting type3. The number of ModernTCN blocks was varied between 2, 3, and 4, with 3 being the best choice.

Speech Commands (Raw): In the raw feature setup for Speech Commands, patch size and stride options of 1,1; 16,8; 32,16; and 64,32 were considered, with 32,16 being selected. Dimensions of 64, 128, 256, and 512 were evaluated, with 512 being optimal. Kernel sizes of 13, 31, 51, and 71 were tested, with 71 being chosen. The learning rate scheduling options were the same as the MFCC setup, with type3 being selected. The number of ModernTCN blocks was set to 3 after testing 2, 3, and 4.

PhysioNet: For the PhysioNet task, patch size and stride options of 1,1; 8,4; and 16,8 were evaluated, with 1,1 being selected. Dimensions of 16, 32, 64, and 128 were considered, with 32 being chosen. Kernel sizes of 13, 31, 51, and 71 were tested, with 31 being optimal. The learning rate scheduling options were the same as previous setups, with type3 being selected. The number of ModernTCN blocks was set to 4 after testing 2, 3, and 4.

Character Trajectories: In the Character Trajectories task, patch size and stride options of 1,1; 8,4; and 16,8 were considered, with 1,1 being selected. Dimensions of 16, 32, 64, and 128 were evaluated, with 32 being optimal. Kernel sizes of 13, 31, 51, and 71 were tested, with 13 being chosen. The learning rate scheduling options were the same as previous setups, with type3 being selected. The number of ModernTCN blocks was set to 2 after testing 2, 3, and 4.

TSB-AD Anomaly Detection (Multivariate): For the multivariate TSB-AD anomaly detection task, window sizes of 32, 96, and 192 were considered, with 96 being selected. Learning rates of 1e-3, 1e-4, and 1e-5 were tested, with 0.001 being optimal. Large sizes of 13, 31, 51, and 71 were evaluated, with 13 being chosen.

TSB-AD Anomaly Detection (Univariate): In the univariate TSB-AD anomaly detection task, the same parameters as the multivariate setup were used. The selected parameters were a window size of 96, a learning rate of 0.001, and a large size of 13.

Short Term Forecasting on ETTh1: For short term forecasting on ETTh1, input sequence lengths of 6x1, 6x2, and 6x3 for a prediction length of 6 were evaluated, with 6x3 being selected. Kernel sizes of 7 and 13 were tested, with 13 being chosen. For a prediction length of 12, input sequence lengths of 12x1, 12x2, and 12x3 were considered, with 12x3 being optimal. Kernel sizes of 7, 13, and 31 were tested, with 13 being selected. For a prediction length of 18, input sequence lengths of 18x1, 19x2, and 18x3 were evaluated, with 18x3 being chosen. Kernel sizes of 7, 13, and 31 were tested, with 13 being optimal.

Short Term Forecasting on ETTm1: In the short term forecasting task on ETTm1, input sequence lengths of 6x1, 6x2, and 6x3 for a prediction length of 6 were considered, with 6x3 being selected. Kernel sizes of 7 and 13 were tested, with 13 being chosen. For a prediction length of 12, input sequence lengths of 12x1, 12x2, and 12x3 were evaluated, with 12x3 being optimal. Kernel sizes of 7, 13, and 31 were tested, with 13 being selected. For a prediction length of 18, input sequence lengths of 18x1, 19x2, and 18x3 were considered, with 18x3 being chosen. Kernel sizes of 7, 13, and 31 were evaluated, with 13 being optimal.

A.2 Anomaly Detection Datasets of TSB-U-AD and TSB-M-AD

We extend the evaluation to TSB-AD (Liu & Paparrizos, 2024), a comprehensive benchmark for anomaly detection. This benchmark consists of 1070 time series from a diverse collection of 40 datasets, including 17 univariate and 13 multivariate datasets. For detailed characteristics, see Table 11.

Table 11: Summary characteristics of 40 datasets included in TSB-AD. '-' in the 2nd column indicates this dataset is transformed from the multivariate dataset. The 'Category' column indicates whether the datasets feature point anomalies (P) or sequence anomalies (Seq). Adapted from Liu & Paparrizos (2024)

Name	# TS Collected	# TS Curated	Avg Dim	Avg TS Len	Avg # Anomaly	Avg Anomaly Len	Anomaly Ratio	Category
TSB-AD-U								
UCR Wu & Keogh (2021)	250	228	1	67818.7	1	198.9	0.6%	P&Seq
NAB Ahmad et al. (2017)	58	28	1	5099.7	1.6	370.1	10.6%	Seq
YAHOO Laptev et al. (2015)	367	259	1	1560.2	5.5	2.5	0.6%	P&Seq
IOPS IOP	58	17	1	72792.3	25.6	48.7	1.3%	Seq
MGAB Thill et al. (2020)	10	9	1	97777.8	9.7	20.0	0.2%	Seq
WSD Zhang et al. (2022)	210	111	1	17444.5	5.1	25.4	0.6%	Seq
SED Boniol et al. (2021)	6	3	1	23332.3	14.7	64.0	4.1%	Seq
TODS Lai et al. (2021)	15	15	1	5000.0	97.3	18.7	6.3%	P&Seq
NEK Si et al. (2024)	48	9	1	1073.0	2.9	51.1	8.0%	P&Seq
Stock Tran et al. (2016)	90	20	1	15000.0	1246.9	1.1	9.4%	P&Seq
Power Keogh et al. (2007)	1	1	1	35040.0	4	750	8.5%	Seq
Daphnet (U) Bachlin et al. (2009)	-	1	1	38774.0	6	384.3	5.9%	Seq
CATSv2 (U) Fleith (2023)	-	1	1	300000.0	19.0	778.9	4.9%	Seq
SWaT (U) Mathur & Tippenhauer (2016)	-	1	1	419919.0	27.0	1876.0	12.1%	Seq
LTDB (U) Goldberger et al. (2000)	-	9	1	99700.0	127.5	144.5	18.6%	Seq
TAO (U) TAO	-	3	1	10000.0	838.7	1.1	9.4%	P&Seq
Exathlon (U) Jacob et al. (2021)	-	32	1	44075.8	3.1	1577.3	11.0%	Seq
MITDB (U) Goldberger et al. (2000)	-	8	1	631250.0	68.7	451.9	4.2%	Seq
MSL (U) Hundman et al. (2018)	-	9	1	3492.0	1.3	130.0	5.8%	Seq
SMAP (U) Hundman et al. (2018)	-	19	1	7700.2	1.2	210.1	2.8%	Seq
SMD (U) Su et al. (2019a)	-	38	1	24207.7	2.4	173.7	2.0%	Seq
SVDB (U) Greenwald (1990)	-	20	1	171380.0	36.4	292.5	3.6%	Seq
OPP (U) Roggen et al. (2010)	-	29	1	16544.8	1.4	653.4	6.4%	Seq
TSB-AD-M								
GHL Filonov et al. (2016)	48	25	19	199001.0	2.2	1035.2	1.1%	Seq
Daphnet Bachlin et al. (2009)	17	1	9	38774.0	6.0	384.3	5.9%	Seq
Exathlon Jacob et al. (2021)	72	27	21	60878.4	4.3	1373.3	9.8%	Seq
Genesis von Birgelen & Niggemann (2018)	1	1	18	16220.0	3.0	16.7	0.3%	Seq
OPP Roggen et al. (2010)	24	8	248	17426.75	1.4	394.3	4.1%	Seq
SMD Su et al. (2019a)	28	22	38	25466.4	8.9	112.8	3.8%	Seq
SWaT Mathur & Tippenhauer (2016)	4	2	59	207457.5	16.5	1093.6	12.7%	Seq
PSM Abdulaal et al. (2021)	1	1	25	217624.0	72.0	338.6	11.2%	P&Seq
SMAP Hundman et al. (2018)	54	27	25	7855.9	1.3	196.3	2.9%	Seq
MSL Hundman et al. (2018)	27	16	55	3119.4	1.3	111.7	5.1%	Seq
CreditCard Sharafaldin et al. (2018)	1	1	29	284807.0	465.0	1.1	0.2%	P&Seq
GECCO Moritz et al. (2018)	1	1	9	138521.0	51.0	33.8	1.2%	Seq
MITDB Goldberger et al. (2000)	48	13	2	336153.8	15.2	1846.8	2.7%	Seq
SVDB Greenwald (1990)	78	31	2	207122.6	68.3	268.2	4.8%	Seq
LTDB Goldberger et al. (2000)	7	5	2	100000.0	105.0	134.4	15.5%	Seq
CATSv2 Fleith (2023)	10	6	17	240000.0	11.5	811.6	3.7%	Seq
TAO TAO	45	13	3	10000.0	788.2	1.1	8.7%	P&Seq

B Visualizing the ERF

Formally, let $I(n \times m \times l)$ be the input time series, where n is the number of samples, m is the number of variables, and l is the input sequence length. Let $F(n \times m \times d \times l')$ be the final output feature map, we desire to measure the contributions of every time point on I to the central points of every channel on F, *i.e.*, $\mathrm{F}_{:,:,:,l'/2}$, which can be simply implemented via taking the derivatives of $\mathrm{F}_{:,:,:,l'/2}$ to I with the auto-grad mechanism. Concretely, we sum up the central points, take the derivatives to the input as the time-wise contribution scores and remove the negative parts (denoted by P). Then we aggregate the entries across all the examples and the input variables, and take the logarithm for better visualization. Formally, the aggregated contribution score matrix $A(l)$ is given by

$$P = \max\left(\frac{\partial(\sum_i^n \sum_j^m \sum_k^d \mathrm{F}_{i,j,k,l'/2})}{\partial \mathrm{I}}, 0\right), \tag{1}$$

$$A = \log_{10}\left(\sum_i^n \sum_j^m P_{i,j,:} + 1\right). \tag{2}$$

Then we respectively rescale A of each model to $[0, 1]$ via dividing the maximum entry for the comparability across models.

(a) ModernTCN (b) Our Method

Figure 4: ERF visualizations from ModernTCN and our method for the same setup.

Following the mentioned methodology (from Kim et al. (2023) and Ding et al. (2022a)) as adapted in Luo & Wang (2024), we explored the ERF visualizations for the ETTh1 forecasting task with an input sequence length of 336. The ModernTCN approach claims to adapt these methodologies for 1D sequences; however, the specifics of their adaptation are not detailed, nor is the code available in their official repository. To establish a reference point, we visualized the same model under the same setup. As shown in Figure 4, the ERF visualization from ModernTCN exhibits more colored areas compared to our method. Such differences might be attributed to the color scheme used. In our visualizations, we clearly indicate the range of colors corresponding to values between [0,1], whereas ModernTCN does not. Additionally, in our plots, not all black areas represent zero gradient contribution; rather, they are very close to zero relative to the maximum gradient contribution due to normalization. Upon closer inspection, the black regions near the center in our visualization reveal a slight red hue, indicating a minimal gradient contribution.

Nevertheless, as depicted in Figure 2, the global convolutions-based method demonstrates a significantly greater ability to enlarge the ERF compared to ModernTCN.

C Full Results

C.1 Long-Term Forecasting

Table 12: Long-term forecasting results. "Reported": scores from ModernTCN paper. "Rerun": our reproduced results. "N.D.L.": No Drop Last, results with test loader not dropping incomplete last batches. "Len" indicates prediction length in time steps (96, 192, 336, and 720 for most datasets; 24, 36, 48, and 60 for ILI). Lower MSE/MAE indicates better performance.

		ETTh1		ETTh2		ETTm1		ETTm2		Electricity		Weather		Traffic		Exchange		ILI	
Model	**Len**	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Reported	96	0.368	0.394	0.263	0.332	0.292	0.346	0.166	0.256	0.129	0.226	0.149	0.200	0.368	0.253	0.080	0.196	1.347	0.717
	192	0.405	0.413	0.320	0.374	0.332	0.368	0.222	0.293	0.143	0.239	0.196	0.245	0.379	0.261	0.166	0.288	1.250	0.778
	336	0.391	0.412	0.313	0.376	0.365	0.391	0.272	0.324	0.161	0.259	0.238	0.277	0.397	0.270	0.307	0.398	1.388	0.781
	720	0.450	0.461	0.393	0.433	0.416	0.417	0.351	0.381	0.191	0.286	0.314	0.334	0.440	0.296	0.656	0.582	1.774	0.868
Rerun	96	0.369	0.394	0.264	0.333	0.297	0.348	0.169	0.256	0.131	0.227	0.150	0.204	0.371	0.256	0.081	0.197	1.348	0.718
	192	0.406	0.414	0.318	0.373	0.334	0.370	0.227	0.299	0.146	0.243	0.195	0.247	0.384	0.267	0.167	0.290	1.448	0.820
	336	0.392	0.412	0.314	0.376	0.371	0.395	0.276	0.329	0.166	0.264	0.237	0.283	0.404	0.273	0.314	0.402	1.389	0.781
	720	0.450	0.461	0.394	0.432	0.419	0.419	0.349	0.390	0.194	0.288	0.315	0.335	0.446	0.298	0.659	0.583	1.775	0.868
N.D.L.	96	0.376	0.397	0.274	0.340	0.295	0.346	0.168	0.255	0.133	0.228	0.148	0.203	0.382	0.265	0.081	0.198	1.935	0.818
	192	0.410	0.417	0.335	0.384	0.334	0.370	0.226	0.299	0.152	0.248	0.194	0.246	0.395	0.270	0.168	0.290	2.109	0.951
	336	0.435	0.433	0.368	0.412	0.374	0.395	0.273	0.327	0.174	0.268	0.244	0.284	0.408	0.277	0.307	0.399	2.008	0.895
	720	0.456	0.466	0.410	0.442	0.420	0.419	0.351	0.391	0.198	0.290	0.318	0.336	0.452	0.309	0.814	0.679	1.949	0.904

C.2 Imputation

Table 13: Imputation task results with varying percentages of missing data (12.5%, 25%, 37.5%, 50%). "Original": reproduced results. "N.D.L.": No Drop Last, results with test loader not dropping incomplete last batches. "Miss %" indicates percentage of missing data. Lower MSE/MAE indicates better performance. All settings from original paper's repository.

Model		ECL		Weather		ETTh1		ETTh2		ETTm1		ETTm2	
	Miss %	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Original	12.5	0.0588	0.1702	0.0241	0.0394	0.0338	0.1263	0.0378	0.1220	0.0158	0.0831	0.0175	0.0765
	25.0	0.0694	0.1812	0.0250	0.0410	0.0436	0.1427	0.0398	0.1271	0.0184	0.0887	0.0186	0.0800
	37.5	0.0805	0.1956	0.0272	0.0443	0.0540	0.1571	0.0434	0.1342	0.0219	0.0961	0.0207	0.0851
	50.0	0.0842	0.1986	0.0314	0.0523	0.0675	0.1747	0.0481	0.1424	0.0270	0.1064	0.0231	0.0914
N.D.L.	12.5	0.0588	0.1702	0.0239	0.0396	0.0338	0.1263	0.0378	0.1220	0.0158	0.0831	0.0175	0.0761
	25.0	0.0694	0.1812	0.0252	0.0412	0.0436	0.1427	0.0398	0.1271	0.0184	0.0888	0.0187	0.0803
	37.5	0.0805	0.1956	0.0273	0.0442	0.0540	0.1571	0.0434	0.1342	0.0219	0.0961	0.0205	0.0848
	50.0	0.0842	0.1986	0.0313	0.0521	0.0675	0.1747	0.0481	0.1424	0.0270	0.1064	0.0230	0.0910

D Drop Last Trick

The "Drop Last Trick" refers to discarding the last batch if it contains fewer instances than the batch size during evaluation. Table 14 examines how this affects model performance across different datasets. Our analysis reveals that significant changes in MSE and MAE primarily occur when more than 5% of the test set is dropped, as seen with ETTh1, ETTh2, Exchange (720), and ILI datasets. In most cases, performance metrics increase (positive Δ MSE/MAE values) when all data points are included, indicating worse performance as lower values are better for these metrics. This suggests that models may have been implicitly tuned for the smaller test sets. While most datasets show performance degradation when including all data points, cases like Exchange (336) show improvement, which aligns with the fact that dropping data points can lead to either better or worse performance depending on the characteristics of the excluded points. This indicates that this faulty setup significantly affects the results, especially when a high percentage of the test set is dropped due to high batch size and/or low test set size.

Table 14: Analysis of the "Drop Last Trick" impact across datasets. For each dataset, we show test set size, batch size, excluded data points, and performance differences. Numbers in parentheses indicate prediction length for forecasting tasks. Δ MSE/MAE shows the difference (No Drop Last - Rerun), with positive values indicating performance degradation when all data points are included.

Task	Dataset	Test Set Size	Batch Size	Dropped Points	Dropped Points (%)	Δ MSE	Δ MAE
Long-term Forecasting	ETTm1	11,521	512	257	2.23%	+0.001	0.000
Long-term Forecasting	ETTm2	11,521	512	257	2.23%	-0.001	-0.001
Long-term Forecasting	ETTh1	2,881	512	321	11.10%	+0.015	+0.008
Long-term Forecasting	ETTh2	2,881	512	321	11.10%	+0.025	+0.015
Long-term Forecasting	Electricity	5,261	32	13	0.25%	+0.004	+0.003
Long-term Forecasting	Traffic	3,509	32	21	0.60%	+0.009	+0.006
Long-term Forecasting	Weather (96)	10,540	256	44	0.42%	-0.002	-0.001
Long-term Forecasting	Weather (192)	10,540	256	44	0.42%	-0.001	-0.001
Long-term Forecasting	Weather (336)	10,540	512	300	2.85%	+0.007	+0.001
Long-term Forecasting	Weather (720)	10,540	512	300	2.85%	+0.003	+0.001
Long-term Forecasting	Exchange (96)	1,422	128	14	0.98%	0.000	+0.001
Long-term Forecasting	Exchange (192)	1,422	128	14	0.98%	+0.001	0.000
Long-term Forecasting	Exchange (336)	1,422	512	398	27.99%	-0.007	-0.003
Long-term Forecasting	Exchange (720)	1,422	512	398	27.99%	+0.155	+0.096
Long-term Forecasting	ILI	170	32	10	5.88%	+0.510	+0.095
Imputation	ETTm1	11,521	16	1	0.01%	0.000	0.000
Imputation	ETTm2	11,521	16	1	0.01%	0.000	0.000
Imputation	ETTh1	2,881	16	1	0.03%	0.000	0.000
Imputation	ETTh2	2,881	16	1	0.03%	0.000	0.000
Imputation	Electricity	5,261	16	13	0.25%	0.000	0.000
Imputation	Weather	10,540	16	12	0.11%	0.000	0.000

E Excluded Results

The results presented in this section were excluded from our main analysis due to methodological concerns regarding validation procedures. As detailed in the Experimental Setup section, both the M4 dataset for

short-term forecasting and the UEA dataset for classification are provided with only train and test splits, with no designated validation set. Upon reviewing the source code, we identified that the test set was used for validation in both cases, introducing data leakage as the model gains indirect exposure to the test data during training through early stopping and hyperparameter selection.

A methodologically sound approach would involve creating a validation split from the training data. For instance, N-BEATS (Oreshkin et al., 2020) addresses this issue with M4 by first creating a validation split from the training set, finding optimal training settings on that validation set, and then training on the full training set using these optimal settings. However, we chose not to implement this approach for several reasons:

First, many of these datasets are already small for deep learning approaches. For example, M4-Weekly (359 samples), M4-Hourly (414 samples), and several UEA classification datasets like EthanolConcentration (261 samples) and Heartbeat (204 samples) have limited training samples to create a validation split from.

Second, implementing a proper validation approach like N-BEATS (Oreshkin et al., 2020) for ModernTCN would yield worse results (at best the same results) than the reported, as the training is on full training set in both cases. On top of that, those reported results are already worse than the reported results in N-BEATS Oreshkin et al. (2020).

Rather than allocating resources to rerun these experiments with proper validation procedures, we extended our study to new datasets such as multivariate short-term forecasting on ETT, time series classification on Speech Commands, and the PhysioNet 2019 sepsis challenge. These extensions allowed for more methodologically sound comparisons and a more comprehensive evaluation of ModernTCN's capabilities across diverse time series tasks.

The tables below present the reproduced results for completeness, but we emphasize that these should be interpreted with caution due to the methodological concerns outlined above.

E.1 M4 Dataset Results

Table 15: Short-term forecasting task. Results are weighted averaged from several datasets under different sample intervals. Lower metrics indicate better performance. Reported refers to the scores reported in the ModernTCN paper. Rerun refers to the scores obtained by rerunning the model with the settings specified in the source code.

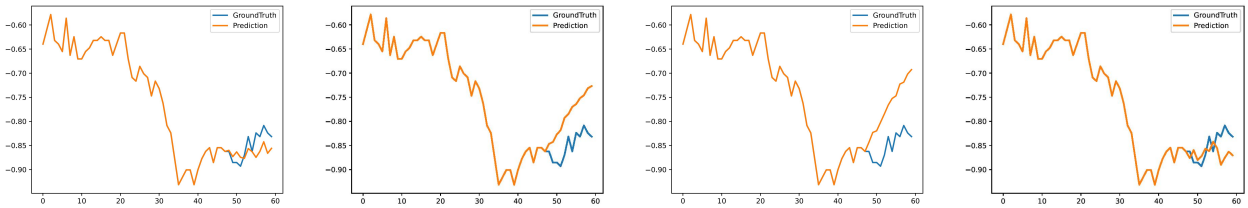
	Yearly			Quarterly			Monthly			Others			Weighted Average		
Model	SMAPE	MASE	OWA	SMAPE	MASE	OWA	SMAPE	MASE	OWA	SMAPE	MASE	OWA	SMAPE	MASE	OWA
Reported	13.226	2.957	0.777	9.971	1.167	0.878	12.556	0.917	0.866	4.715	3.107	0.986	11.698	1.556	0.838
Rerun	13.231	2.957	0.777	10.001	1.170	0.881	12.598	0.920	0.868	4.835	3.175	1.009	11.732	1.561	0.841

E.2 UEA Dataset Results

Table 16: Classification task. Accuracy metric is used. Reported refers to the scores reported in the ModernTCN paper. Rerun refers to the scores obtained by rerunning the model with the settings specified in the source code. Datasets: EthanolConcentration (EC), FaceDetection (FD), Handwriting (HW), Heartbeat (HB), JapaneseVowels (JV), PEMS-SF (PS), SelfRegulationSCP1 (SR1), SelfRegulationSCP2 (SR2), SpokenArabicDigits (SAD), UWaveGestureLibrary (UWL).

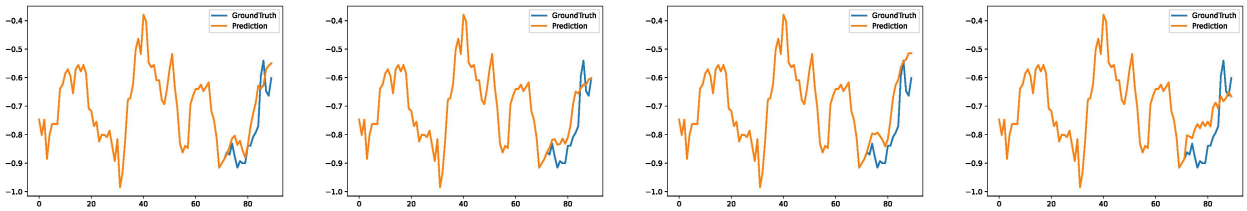
	EC	FD	HW	HB	JV	PS	SR1	SR2	SAD	UWL	Averaged
Reported	0.363	0.708	0.306	0.772	0.988	0.891	0.934	0.603	0.987	0.867	0.742
Rerun	0.319	0.687	0.284	0.771	0.981	0.832	0.928	0.617	0.981	0.859	0.726

F Short-Term Forecasting on ETT



(a) ModernTCN (b) PatchTST (c) DLinear (d) TimesNet

Figure 5: Short-term forecasting on ETTm1 with prediction length 12 and input length 48.



(a) ModernTCN (b) PatchTST (c) DLinear (d) TimesNet

Figure 6: Short-term forecasting on ETTh1 with prediction length 18 and input length 72.