SQUAD: Multi-Scale LSTM with Smooth Quadratic Function for Long-Term Time Series Forecasting

Anonymous submission

Abstract

In recent years, the growing attention to long-term time series forecasting has prompted considerable research efforts. While recent models have shown significant performance, the effective and comprehensive exploration of temporal features remains a challenge to be addressed. To tackle this challenge, we propose an elegant and effective model called Multi-scale LSTM (MSLSTM). The MSLSTM model excels at capturing information at different temporal scales, enabling fusion and prediction of features from diverse scales and achieving remarkable forecasting performance. Additionally, we introduce a robust and dedicated loss function, namely the Smooth Rational Function (SQF), specifically tailored for multivariate time series prediction. Through rigorous mathematical analysis, we demonstrate its superior noise resilience compared to the commonly used Mean Squared Error (MSE) loss function. By combining the proposed model with the novel loss function, we conduct extensive experiments on eight real-world datasets, surpassing the state-of-the-art models. Moreover, we evaluate the applicability of the Multi-scale approach and the SQF as plug-and-play modules in enhancing the results of other models for multivariate time series forecasting. Our findings highlight the efficacy of these modules in achieving superior prediction accuracy.

1 Introduction

Time series prediction is a crucial task across diverse fields, including but not limited to power (Kardakos et al. 2013), transportation (Kadiyala and Kumar 2014), weather (Karevan and Suykens 2020), and healthcare (Morid, Sheng, and Dunbar 2023). This prediction task typically involves multiple indicators that need to be forecasted accurately. In this paper, we delve into the challenge of multivariate time series prediction and demonstrate the effectiveness of our approach in various domains with satisfactory performance.

In the past, ARIMA models (Bartholomew 1971) were widely used for multivariate time series prediction, but they are limited by the requirement for stationary data and linear relationships. As a result, machine learning algorithms such as SVM (Hearst et al. 1998) and Xgboost (Chen et al. 2015) gained attention for handling nonlinear and non-stationary data in real-world prediction tasks. With advancements in deep learning, various RNN-based models (Dey and Salem 2017; Graves and Graves 2012) were explored, including GRU and LSTM. GRU performed well on short sequences, while LSTM demonstrated effectiveness in capturing long-term dependencies. However, LSTM comes with higher computational complexity due to more parameters and slower training speed. As the length of the input sequence increases, the inference speed of LSTM decreases, rendering it less effective.

In recent times, the Transformer model (Vaswani et al. 2017) and its variants, such as LogTrans (Li et al. 2019), Pyraformer (Liu et al. 2021), Informer (Zhou et al. 2021), Autoformer (Wu et al. 2021), and FEDformer (Zhou et al. 2022), have shown remarkable success in multivariate time series tasks. Although many of these variants improved the attention mechanism to reduce computational complexity, challenges remained in addressing complexity reduction at the model's input level. PatchTST (Nie et al. 2023), inspired by Vision Transformer (ViT) (Dosovitskiy et al. 2020), offers a promising solution by dividing the sequence into multiple patches of the same scale and using the Transformer for prediction. However, using a single-scale patch as input limits the model's ability to exploit multi-resolution information effectively, leading to suboptimal performance.

In the context of loss function research for time series prediction, common metrics like Mean Squared Error (MSE) (Bao and Ullah 2007) and Dynamic Time Warping (DTW) (Cuturi and Blondel 2017) have their limitations. MSE is sensitive to outliers, while DTW overlooks time delays. DILATE (Le Guen and Thome 2019) attempted to address this by combining shape and time loss functions with weighted adjustments, yet it may still struggle with non-stationary, nonlinear, and noisy real-world data, lacking accuracy in measuring data nonlinearity and showing limited noise resilience.

In this work, we propose a novel smooth loss function based on the rational quadratic function to tackle the challenges associated with time series prediction. This new approach aims to enhance model accuracy and robustness in the presence of nonlinearity and noise.

Motivated by the above, we propose a novel model called Multi-scale LSTM (MSLSTM) that leverages LSTM as a backbone for multi-scale time series feature extraction. The MSLSTM model effectively combines multi-scale features to enhance the performance of temporal prediction. Additionally, we introduce a novel loss function for multivariable time series prediction, named Smooth Quadratic

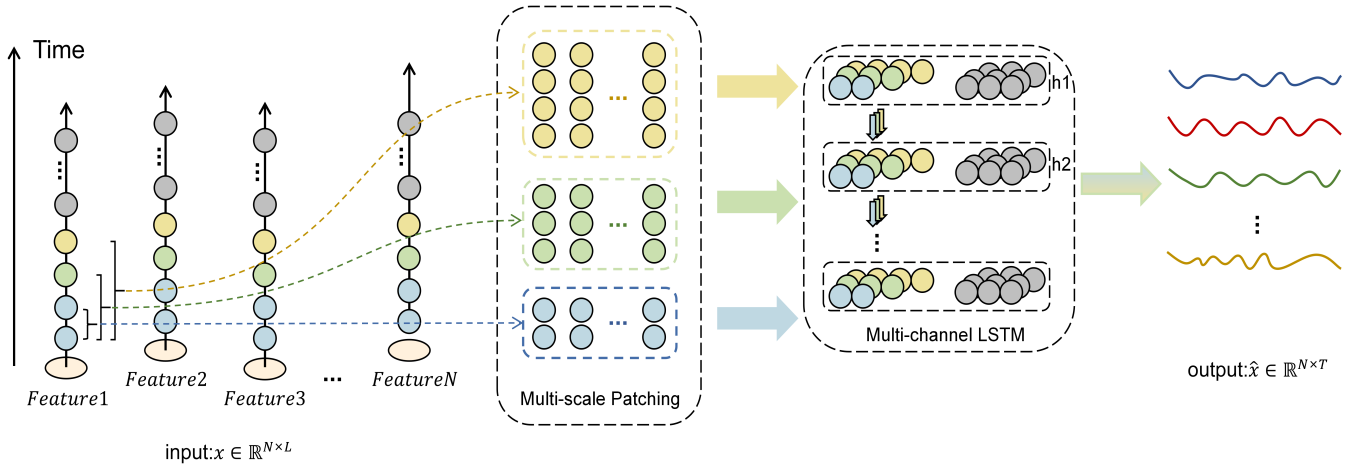


Figure 1: Overview of MSLSTM Model

Loss (SQF). The SQF exhibits improved efficiency in reducing sensitivity to outlier values and demonstrates enhanced robustness against noise interference. the contributions of this paper are threefold:

- We propose a novel loss function, named Smooth Quadratic Function (SQF), for time series forecasting. Our approach effectively maps errors to a high-dimensional space, thereby enhancing the performance in capturing complex patterns within the data and improving robustness against noise interference.
- We propose a simple multivariate time series forecasting framework, named Multi-scale LSTM (MSLSTM), which employs multi-scale patches to extract features at multiple resolutions. An overview of the proposed model is shown in Figure 1.
- We conducted extensive experiments to demonstrate the superiority of our proposed method. Specifically, we compared our approach against state-of-the-art models from the past several years on eight datasets, and our method outperformed nearly all other approaches on each dataset.

2 Problem definition

A data point in time series forecasting contains historical data X and the subsequent part of the time series Y (also called ground truth). Time series forecasting aims to predict future time series based on historical data. Formally speaking, given the historical time series with window L: $\boldsymbol{X} = [\boldsymbol{x}^1, \ldots, \boldsymbol{x}^L]$, the model is required to predict future values with length T: $\hat{\boldsymbol{X}} = [\hat{\boldsymbol{x}}^{L+1}, \ldots, \hat{\boldsymbol{x}}^{L+T}]$. The goal of a time series forecasting model is to minimize the difference between the prediction and the ground truth.

Without loss of generality, the input time series $X \in \mathbb{R}^{N\times L}$ contains multiple variables, denoted by

$$X = [\boldsymbol{x}^1, \ldots, \boldsymbol{x}^L] = \begin{bmatrix} x_1^1 & x_1^2 & \ldots & x_1^L \\ x_2^1 & x_2^2 & \ldots & x_2^L \\ \vdots & \vdots & \vdots & \vdots \\ x_N^1 & x_N^2 & \ldots & x_N^L \end{bmatrix}, \quad (1)$$

where N stands for the number of variables. Similarly, the desired output multivariate $\hat{X} \in \mathbb{R}^{N\times T}$ is

$$\hat{X} = [\hat{\boldsymbol{x}}^{L+1}, \ldots, \hat{\boldsymbol{x}}^{L+T}] = \begin{bmatrix} \hat{x}_1^{L+1} & \hat{x}_1^{L+2} & \ldots & \hat{x}_1^{L+T} \\ \hat{x}_2^{L+1} & \hat{x}_2^{L+2} & \ldots & \hat{x}_2^{L+T} \\ \vdots & \vdots & \vdots & \vdots \\ \hat{x}_N^{L+1} & \hat{x}_N^{L+2} & \ldots & \hat{x}_N^{L+T} \end{bmatrix}. \quad (2)$$

It is noted that the number of variables (i.e., N) in the prediction equals the one in the input. According to the above notation, the superscript represents the index of time steps and the subscript denotes the index of the variable in the multivariate. For notation simplicity, we use $\boldsymbol{x}_n \in \mathbb{R}^L$ to indicate the time series of the n-th variable. That is, the input X could also be represented by $[\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N]^T$.

3 Related work

Deep models for time series forecasting

Time series prediction research has witnessed significant advancements, driven by its wide-ranging applications. Traditional recurrent neural networks (RNNs) (Medsker and Jain 2001) are effective for sequential data but struggle with long sequences due to issues like vanishing and exploding gradients. LSTM (Graves and Graves 2012) and GRU (Dey and Salem 2017) models have emerged as improved RNN-based solutions, demonstrating robustness in capturing long-term dependencies and excelling in multivariate time series prediction tasks.

Advancements in RNN-based models continue with LSTNet (Lai et al. 2018), which combines LSTM, CNNs, and self-attention mechanisms to address long-term dependencies and computational concerns. TCN (Bai, Kolter, and Koltun 2018) introduces a novel approach with varying lengths of kernels and a sequence of convolutional neural networks for feature extraction.

Recently, Transformer models, successful in natural language processing (Vaswani et al. 2017; Devlin et al. 2018; Radford et al. 2018), have been extended to time series forecasting. LogTrans (Li et al. 2019) uses convolutional self-attention with causal convolution, incorporating logsparse masks for computational efficiency. Informer (Zhou et al.

2021) selects dominant queries to achieve similar improvements. Pyraformer (Liu et al. 2021) uses hierarchical pyramid attention, AutoFormer (Wu et al. 2021) adopts a seasonal decomposition architecture, and FEDFormer (Zhou et al. 2022) utilizes attention in the frequency domain.

Additionally, there are models like TimesNet (Wu et al. 2023), which transforms time series into 2D tensors for easier modeling, and MICN (Wang et al. 2023), efficiently combining local and global features for accurate long-term forecasting.

However, concerns have been raised about the suitability of Transformers for time series prediction, as seen in the Dlinear (Zeng et al. 2023) model. In contrast, the PatchTST (Nie et al. 2023) model, inspired by ViT (Dosovitskiy et al. 2020) in visual tasks, outperforms Dlinear by decomposing time series data into patches before inputting them into a Transformer model.

In our work, we propose an LSTM model based on multiple patches, extracting features from patches of different scales to leverage LSTM's natural ability in capturing temporal patterns, resulting in promising results for multivariate time series prediction tasks.

Loss function of time series forecasting

While traditional loss functions such as MSE and L_p norm have been widely used in time series forecasting, they do not fully capture the temporal continuity and correlation of the data. To address this limitation, researchers have explored alternative metrics such as differentiable, approximated dynamic time warping (DTW) (Cuturi and Blondel 2017), which can better model the temporal dynamics of time series data. However, using DTW as a loss function may result in ignoring changes in temporal localization, leading to inaccurate predictions. To overcome this limitation, a new training metric is proposed DILATE (Le Guen and Thome 2019) that timely catches sudden changes in non-stationary signals by penalizing the temporal distortion index (TDI). While DILATE demonstrates improved performance over traditional loss functions, it can lose its advantage with complex data and exhibit disadvantages during the early training stage. In this work, we propose a smooth loss function based on a rational quadratic function to improve the prediction effect.

4 Methodology

To predict the multivariate time series, we first introduce a multi-scale LSTM module to efficiently extract the temporal feature underlying the multivariate dynamics. Then we elaborate on the smooth quadratic loss function to eliminate the distraction of the inevitable noises in time series.

Multi-Scale LSTM

The extraction of information at various scales is of paramount importance in time series prediction. Long-scale information facilitates the capture of broad patterns and changes over extended periods, which is highly valuable for forecasting long-term trends or cycles. Conversely, short-scale information enables the capture of fine-grained details and localized variations. By integrating information from different scales, predictive models can reap the benefits of a comprehensive understanding of the dynamics underlying the time series, resulting in more accurate predictions. To this end, we first propose a multi-scale patching operation to build the encoding of the time series.

In addition, multi-scale information could close the gap between LSTM and the transformer model in processing the long-length time series. Considering that LSTM excels in modeling multivariate dynamics and shares parameters across different steps, we integrate multi-scale patching and LSTM to build the encoder of the multivariate time series.

Multi-scale patching In time series, a single step of the variables could hardly have meaningful information in the temporal process. Therefore, inspired by (**?**), we introduce a patching operation to capture the locality of the semantic information in the time series. As shown in Figure 2, the patching operation aggregates several time steps into a patch of the local semantic context of the time series. Formally speaking, the i-th patch of the n-th variable is denoted as $\boldsymbol{p}_{n,i}$, given by

$$\boldsymbol{p}_{n,i} = \boldsymbol{X}_{n,(i-1)*S^{(s)}:(i-1)*S^{(s)}+S^{(p)}} \tag{3}$$

$$= [\boldsymbol{x}_{n,(i-1)*S^{(s)}}, \cdots, x_{n,(i-1)*S^{(s)}+S^{(p)}}] \in \mathbb{R}^{S^{(p)}} \tag{4}$$

where $S^{(p)}$ and $S^{(s)}$ stand for the patch scale and the sliding size, respectively. In total, there is $N^{(p)} = \lfloor \frac{L-S^{(p)}}{S^{(s)}} \rfloor + 1$ patches, that is $i \in \{1, 2, \cdots, N^{(p)}\}$. For example, for a single variable $\boldsymbol{x}_1$ with the length L, it could be transformed into $L-2$ patches if the patch scale $S^{(p)}$ is 3 and the sliding size $S^{(s)}$ is 1. Therefore, the patching operation transforms the input time series into multiple patches, represented as $\boldsymbol{P}$.

$$\boldsymbol{P} = \text{Patching}(\boldsymbol{X}, S^{(p)}, S^{(s)}) \in \mathbb{R}^{N \times N^{(p)} \times S^{(p)}} \tag{5}$$

In this way, the time series is divided into multiple local contexts, similar to the discrete words in the natural language.

The above patching operation captures the semantic contexts of time series with a single scale. Next, we adopt K patching scales and patching striding sizes to capture sufficient information underlying the temporal dynamics, which is coined as multi-scale patching.

$$\boldsymbol{P}^{(k)} = \text{Patching}(\boldsymbol{X}, S^{(p,k)}, S^{(s,k)}), k = 1, \cdots, K \tag{6}$$

where $\boldsymbol{P}^{(k)}$ stands for the patching features extracted by the k-th patching scale $S^{(p,k)}$ and patching sliding size $S^{(s,k)}$.

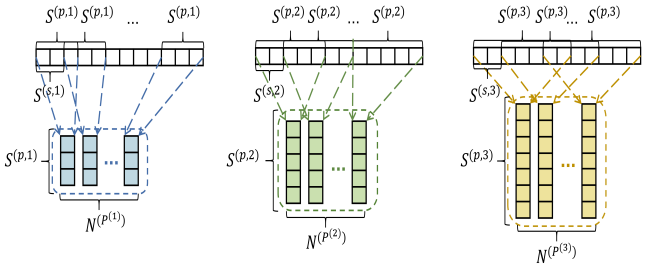


Figure 2: Multi-scale patching.

Multi-scale feature integration As the patches in individual groups capture distinct temporal characteristics, we leverage K LSTM modules to learn the corresponding representations independently, given by

$$\boldsymbol{h}_k = \text{LSTM}_k(\boldsymbol{P}^{(k)}) \in \mathbb{R}^{N \times N^{(p)} \times H}, \tag{7}$$

where $\boldsymbol{h}_k$ denotes the output features of the k-th LSTM module. As the sizes of the multi-scale patches are different, the input sizes of the LSTM modules are adapted to fit the patch sizes. In addition, the hidden sizes of these LSTM modules share the same number of dimensions H. The representations processed by LSTM are further concatenated together as the final features of the historical time series, given by $= [\boldsymbol{h}_1, \boldsymbol{h}_2, \cdots, \boldsymbol{h}_K]$.

Prediction There are two ways to predict the future values of the target feature: Channel Dependent (CD) linear layer and Channel Independent (CI) linear layer (Han, Ye, and Zhan 2023). The CD strategy refers to considering the historical data of all features when predicting the future value of a particular feature, whereas the CI approach only utilizes the historical data of a single channel during prediction. The formula is as follows:

$$\hat{X} = \text{Linear}([\boldsymbol{h}_1, \boldsymbol{h}_2, \cdots, \boldsymbol{h}_K]) \tag{8}$$

Where $\hat{X}$ is the final predicted result.

Smooth Quadratic Function

Due to the unobservable hidden states of the time series, the noises are inevitable in temporal dynamics. Consequently, one of the long-standing problems in time series is how to filter the noises and learn the underlying essential temporal law of the variable changing. The traditional loss functions, such as L1 and L2, do not consider the existence of noises in the ground truth. For example, the gradients under mean square error are linearly increasing upon the prediction error. In this work, we propose a loss function, named smooth quadratic function, that could dynamically adjust the gradient according to the prediction error. The smooth quadratic function is designed based on the rational quadratic kernel function, which calculates the difference between the prediction and the ground truth in a high-dimensional space.

Rational quadratic function Assuming Φ is a nonlinear mapping function from the original feature space to the high-dimensional feature space. The inner product in the kernel space can be defined as follows:

$$\kappa(\hat{\boldsymbol{x}}, \boldsymbol{y}) = < \Phi(\hat{\boldsymbol{x}}), \Phi(\boldsymbol{y}) > \tag{9}$$

where $\hat{\boldsymbol{x}}$ and y stand for the prediction and the label, respetively.

In this study, we propose the use of the rational quadratic function (RQF) as an alternative to the commonly used Gaussian kernel function. The specific form of the rational quadratic function is given by:

$$\kappa(\hat{\boldsymbol{x}}, \boldsymbol{y}) = 1 - \frac{(\hat{\boldsymbol{x}} - \boldsymbol{y})^2}{(\hat{\boldsymbol{x}} - \boldsymbol{y})^2 + c} \tag{10}$$

where c is a hyperparameter. Therefore, the RQF loss for a single data point is written as follows:

$$L_{RQF}(\hat{\boldsymbol{x}}, \boldsymbol{y}) = 1 - \kappa(\hat{\boldsymbol{x}}, \boldsymbol{y}) = \frac{(\hat{\boldsymbol{x}} - \boldsymbol{y})^2}{(\hat{\boldsymbol{x}} - \boldsymbol{y})^2 + c} \tag{11}$$

Besides its complex non-linearity, this loss function is distinguished by its broad scope and rapid computation but is known to be sensitive to the choice of parameters.

The gradient of the rational quadratic loss respective to the prediction $\boldsymbol{y}$ is given by:

$$\frac{\partial L_{RQF}}{\partial \boldsymbol{y}} = \frac{\partial L_{RQF}}{\partial \boldsymbol{e}} = \frac{2c\boldsymbol{e}}{(\boldsymbol{e}^2 + c)^2}, \tag{12}$$

where $\boldsymbol{e}$ stands for the prediction error, i.e., $\boldsymbol{e} = \boldsymbol{y} - \hat{\boldsymbol{x}}$.

Intriguingly, from Figure 3, we observe that the gradient of the rational quadratic function (RQF) loss is nonlinear to the prediction error and insensitive to the outliers in the time series. When the scale of prediction error progressively increases (one possible reason is the growing noises), the amplitude of the RQF gradient is first increasing and then gradually decreasing to zero. This characteristic of the RQF gradient enforces the model first to optimize the prediction error of normal data points. For abnormal, difficult data points or even outliers, it adopts a smaller strength in optimization. Compared with the means square error loss, its gradient respect is linear to the prediction error, which is not agile enough when the noises are non-negligible.

In order to compare the noise immunity of RQF and MSE, we constructed ten time series datasets, adding noise with different standard deviations to the test set, and the details of dataset construction are shown in the Appendix. We evaluated the immunity of the RQF and MSE loss functions using the mean squared error (MSE) as an evaluation metric. As shown in the results from Figure 3, the robustness of the RQF consistently outperformed MSE in terms of noise resistance.

Theoretical analysis. We provide theoretical justifications for the proposed rational quadratic loss function. Below, we investigate how the noises in the ground truth influence the learning of the time series forecasting models.

Definition 1 *The noise in the label of time series forecasting is denoted as ε. The effect of the noise is evaluated by the gradient changes, calculated by $|\frac{f(t+\varepsilon,\hat{\boldsymbol{x}})-f(t,\hat{\boldsymbol{x}})}{f(t,\hat{\boldsymbol{x}})}|$, where t and $\hat{\boldsymbol{x}}$ stand for the ground truth and the prediction, respectively.*

Theorem 1 *Regardless of the distribution of the noise ε in the time-series data, the effect of the noise on the rational RQF loss is always lower than that of MSE loss.*

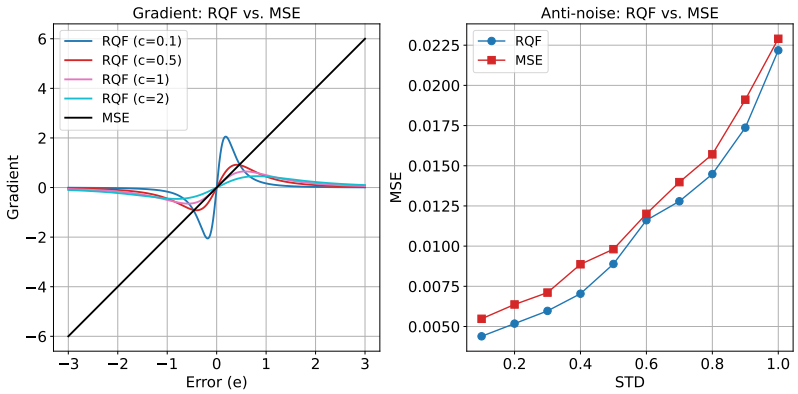


Figure 3: Comparison of RQF and MSE

The proof is in Appendix A.

Theorem 2 *If* $\varepsilon \geq \frac{c^2}{|\hat{\boldsymbol{x}}-t|}$, *we have*

$$\left|\frac{\nabla_\theta RQF(t+\varepsilon,\hat{\boldsymbol{x}})-\nabla_\theta RQF(t,\hat{\boldsymbol{x}})}{\nabla_\theta RQF(t,\hat{\boldsymbol{x}})}\right| \leq \left|\frac{\nabla_\theta MSE(t+\varepsilon,\hat{\boldsymbol{x}})-\nabla_\theta MSE(t,\hat{\boldsymbol{x}})}{\nabla_\theta MSE(t,\hat{\boldsymbol{x}})}\right| \tag{13}$$

To prove it, we simplify the above formula with the fundamental inequality and make some mathematical relaxations based on the preconditions. The detailed proof sketch is in Appendix B.

Remark 1. Theorems 1 and 2 demonstrate that when the ground truth is mixed with noises, both the loss values of the rational quadratic function (RQF) and the corresponding gradients align better with the ground truth than the MSE loss. In other words, the effect of the noises on the RQF loss is always smaller than the MSE loss. Since the above claim is supported in terms of both the loss value and the gradient, we can conclude that the effect of the noises on the learning process will be also smaller if the RQF loss is used.

Smooth quadratic function. We apply the McLaughlin formula (i.e., a particular form of the Taylor series) (Maclaurin 1742) to further investigate the characteristics of RQF, given by

$$L_{RQF}(\hat{\boldsymbol{x}},\boldsymbol{y}) = \sum_{i=1}^{n}(-1)^{i-1}\frac{(\widehat{x}-y)^{2i}}{c^i} + o\left((\widehat{x}-y)^{2n}\right) \tag{14}$$

Notably, RQF is highly nonlinear to measure the prediction errors. In the meanwhile, we observe that the lowest-order term of RQF is quadratic. RQF may reckon without the low-latency information in time series. Thus, we integrate the mean absolute error (MAE) function into RQF to make the loss function smoother.

$$L_{SQF} = \frac{\alpha}{T}\sum_{t=1}^{T}\frac{(y_t-\widehat{x}_t)^2}{(y_t-\widehat{x}_t)^2+c} + \frac{1-\alpha}{T}\sum_{t=1}^{T}|y_t-\widehat{x}_t| \tag{15}$$

where α is the hyperparameter to balance the importance of the two functions.

As another technique to avoid overfitting to the outliers with large amplitudes, an outlier regularization (OR) (Merity, McCann, and Socher 2017) on the predicted time series is proposed. In the outlier regularization, we apply the L1 and L2 regularization to the predicted value of the SQUAD model with small scaling coefficients separately. In this way, OR penalizes the predicted time series that is substantially away from 0, encouraging the activations to remain small and smooth. While most of the regularization techniques are applied to the learnable weights or the layer-wise activations (e.g., layer normalization), the outlier regularization is new and has shown effectiveness in time series forecasting.

In summary, the SQF (Smooth Quadratic Function) loss is the combination of the rational quadratic function (RQF), mean absolute error (MAE), and the outlier regularization (OR), computed by

$$\begin{aligned} L_{SQF} &= \alpha L_{RQF} + (1-\alpha)L_{MAE} + L_{OR} \\ &= \frac{1}{T}\sum_{t=1}^{T}\Big[\alpha\frac{(y_t-\widehat{x}_t)^2}{(y_t-\widehat{x}_t)^2+c} + (1-\alpha)\,|y_t-\widehat{x}_t| \\ &\quad + \beta\,|\widehat{x}_t| + \gamma\widehat{x}_t^2\Big], \end{aligned} \tag{16}$$

where β and γ are the hyperparameters.

5 Experiments

Datasets and competitive methods

We used 8 publicly available datasets commonly used for time series forecasting to evaluate our method, covering various fields, including Weather, Traffic, Electricity, ILI, and ETT (ETTh1, ETTh2, ETTm1, ETTm2) (Zhou et al. 2021; Wu et al. 2021). Most of the time series in these datasets are long-term, requiring the model to capture the historical dynamics. The details of each dataset are shown in Table 2.

We have selected various state-of-the-art (SOTA) models that have emerged in recent years, including Multi-scale Isometric Convolution Network (MICN) (Wang et al. 2023), LogTrans (Li et al. 2019), Pyraformer (Liu et al. 2021), Informer (Zhou et al. 2021), AutoFormer (Wu et al. 2021), Decomposition Linear Model (Dlinear) (Zeng

Models		MSLSTM		PatchTST/42		DLinear		TimesNet		MICN		FEDformer		Autoformer		Informer		Pyraformer		LogTrans	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
Weather	96	**0.148**	**0.185**	0.149	0.198	0.176	0.237	0.172	0.220	0.161	0.229	0.238	0.314	0.249	0.329	0.354	0.405	0.896	0.556	0.458	0.490
	192	**0.190**	**0.227**	0.194	0.241	0.220	0.282	0.219	0.261	0.220	0.281	0.275	0.329	0.325	0.370	0.419	0.434	0.622	0.624	0.658	0.589
	336	**0.243**	**0.268**	0.245	0.282	0.265	0.319	0.280	0.306	0.278	0.331	0.339	0.377	0.351	0.391	0.583	0.543	0.739	0.753	0.797	0.652
	720	0.322	**0.323**	**0.314**	0.334	0.323	0.362	0.365	0.359	0.311	0.356	0.389	0.409	0.415	0.426	0.916	0.705	1.004	0.934	0.869	0.675
Traffic	96	0.380	**0.234**	**0.360**	0.249	0.410	0.282	0.593	0.321	0.519	0.309	0.576	0.359	0.597	0.371	0.733	0.410	2.085	0.468	0.684	0.384
	192	0.400	**0.242**	**0.379**	0.256	0.423	0.287	0.617	0.336	0.537	0.315	0.610	0.380	0.607	0.382	0.777	0.435	0.867	0.467	0.685	0.390
	336	0.412	**0.249**	**0.392**	0.264	0.436	0.296	0.629	0.336	0.534	0.313	0.608	0.375	0.623	0.387	0.776	0.434	0.869	0.469	0.734	0.408
	720	0.444	**0.269**	**0.432**	0.286	0.466	0.315	0.640	0.350	0.577	0.325	0.621	0.375	0.639	0.395	0.827	0.466	0.881	0.473	0.717	0.396
Electricity	96	0.130	**0.219**	**0.129**	0.222	0.140	0.237	0.168	0.272	0.164	0.269	0.186	0.302	0.196	0.313	0.304	0.393	0.386	0.449	0.258	0.357
	192	**0.147**	**0.235**	0.147	0.240	0.153	0.249	0.184	0.289	0.177	0.285	0.197	0.311	0.211	0.324	0.327	0.417	0.386	0.443	0.266	0.368
	336	0.164	**0.253**	**0.163**	0.259	0.169	0.267	0.198	0.300	0.193	0.304	0.213	0.328	0.214	0.327	0.333	0.422	0.378	0.443	0.280	0.380
	720	0.204	**0.287**	**0.197**	0.290	0.203	0.301	0.220	0.320	0.212	0.321	0.233	0.344	0.236	0.342	0.351	0.427	0.376	0.445	0.283	0.376
ILI	24	**1.298**	**0.665**	1.319	0.754	2.215	1.081	2.317	0.934	2.684	1.112	2.624	1.095	2.906	1.182	4.657	1.449	1.420	2.012	4.480	1.444
	36	**1.241**	**0.676**	1.579	0.870	1.963	0.963	1.972	0.920	2.667	1.068	2.516	1.021	2.585	1.038	4.650	1.463	7.394	2.031	4.799	1.467
	48	**1.530**	**0.750**	1.553	0.815	2.130	1.024	2.238	0.940	2.558	1.052	2.505	1.041	3.024	1.145	5.004	1.542	7.551	2.057	4.800	1.468
	60	**1.406**	**0.731**	1.470	0.788	2.368	1.096	2.027	0.928	2.747	1.110	2.742	1.122	2.761	1.114	5.071	1.543	7.662	2.100	5.278	1.560
ETTh1	96	**0.360**	**0.386**	0.370	0.400	0.375	0.399	0.384	0.402	0.390	0.416	0.376	0.415	0.435	0.446	0.941	0.769	0.664	0.612	0.878	0.740
	192	**0.402**	**0.412**	0.413	0.429	0.405	0.416	0.436	0.429	0.441	0.449	0.423	0.446	0.456	0.457	1.007	0.786	0.790	0.681	1.037	0.824
	336	**0.414**	**0.421**	0.422	0.440	0.439	0.443	0.491	0.469	0.506	0.497	0.444	0.462	0.486	0.487	1.038	0.784	0.891	0.738	1.238	0.932
	720	**0.420**	**0.446**	0.447	0.468	0.472	0.490	0.521	0.500	0.640	0.585	0.469	0.492	0.515	0.517	1.144	0.857	0.963	0.782	1.135	0.852
ETTh2	96	**0.274**	**0.328**	**0.274**	0.337	0.289	0.353	0.340	0.374	0.354	0.399	0.332	0.374	0.332	0.368	1.549	0.952	0.645	0.597	2.116	1.197
	192	**0.339**	**0.371**	0.341	0.382	0.383	0.418	0.402	0.414	0.487	0.478	0.407	0.446	0.426	0.434	3.792	1.542	0.788	0.683	4.315	1.635
	336	0.330	**0.373**	**0.329**	0.384	0.448	0.465	0.452	0.452	0.604	0.543	0.400	0.447	0.477	0.479	4.215	1.642	0.907	0.747	1.124	1.604
	720	0.382	**0.415**	**0.379**	0.422	0.605	0.551	0.462	0.468	0.821	0.653	0.412	0.469	0.453	0.490	3.656	1.619	0.963	0.783	3.188	1.540
ETTm1	96	**0.283**	**0.328**	0.293	0.346	0.299	0.343	0.338	0.375	0.315	0.369	0.326	0.390	0.510	0.492	0.626	0.560	0.543	0.510	0.600	0.546
	192	**0.324**	**0.355**	0.333	0.370	0.335	0.365	0.374	0.387	0.365	0.403	0.365	0.415	0.514	0.495	0.725	0.619	0.557	0.537	0.837	0.700
	336	**0.356**	**0.376**	0.369	0.392	0.369	0.386	0.410	0.411	0.409	0.436	0.392	0.425	0.510	0.492	1.005	0.741	0.754	0.655	1.124	0.832
	720	0.424	**0.420**	**0.416**	**0.420**	0.425	0.421	0.478	0.450	0.511	0.502	0.446	0.458	0.527	0.493	1.133	0.845	0.908	0.724	1.153	0.820
ETTm2	96	**0.163**	**0.246**	0.166	0.256	0.167	0.260	0.187	0.267	0.179	0.275	0.180	0.271	0.205	0.293	0.355	0.462	0.435	0.507	0.768	0.642
	192	**0.216**	**0.283**	0.223	0.296	0.224	0.303	0.249	0.309	0.307	0.376	0.252	0.318	0.278	0.336	0.595	0.586	0.730	0.673	0.989	0.757
	336	**0.264**	**0.315**	0.274	0.329	0.281	0.342	0.321	0.351	0.325	0.388	0.324	0.364	0.343	0.379	1.270	0.871	1.201	0.845	1.334	0.872
	720	**0.348**	**0.370**	0.362	0.385	0.397	0.421	0.408	0.403	0.502	0.490	0.410	0.420	0.414	0.419	3.001	1.267	3.625	1.451	3.048	1.328

Table 1: Multivariate long-term forecasting results with MSLSTM. Following the evaluation setting in (), we use prediction lengths $T \in \{24, 36, 48, 60\}$ for ILI dataset and $T \in \{96, 192, 336, 720\}$ for the others. The best results are in **bold** .

et al. 2023), TimesNet (Wu et al. 2023), Frequency Enhanced Decomposed Transformer (FEDFormer) (Zhou et al. 2022), and channel-independent patch time series Transformer (PatchTST) (Nie et al. 2023). In particular, Dlinear and PatchTST are the previously best MLP-based and Transformer-based methods, respectively.

Datasets	Weather	Traffic	Electricity	ILI	ETTh1	ETTh2	ETTm1	ETTm2
# features	21	862	321	7	7	7	7	7
# samples	52696	17544	26304	966	17420	17420	69680	69680
Interval	10 mins	1 hour	1 hour	1 week	1 hour	1 hour	15 mins	15 mins

Table 2: Details of common datasets for benchmark.

Implementation details

In SQUAD, the input sequence length was set to 336, while the prediction length T was chosen from the set $96, 192, 336, 720$ for all datasets. Since the ILI dataset is relative . For the latter, we adjusted the input sequence length to 104, and the prediction length T was selected from $24, 36, 48, 60$. For a comprehensive understanding of the model hyperparameters, we direct readers to the 'More Experimental Details' section in Appendix. This section contains comprehensive information regarding our experimental setup and model hyperparameters, enabling readers to gain deeper insights into the design of our study.

Results

The results of the experiments contain the results of multivariate time series prediction and the results of performing ablation experiments on SQF and applying SQF to other models. More experiments on SQF and MSLSTM can be found in Appendix.

Multivariate Time Series Forecasting We utilized both MSE and MAE metrics to comprehensively evaluate the performance of our model. Table 1 showcases the results of multivariate time series prediction. Notably, our proposed method exhibits superior performance across all evaluation scenarios, particularly excelling in the MAE metric, where it outperformed all baseline approaches for all prediction length settings. Although the MSE metric slightly trailed behind in 34.38% of cases, the differences remained close to the best-performing metric.

Specifically, our proposed method demonstrated substantial reductions in the MAE metric on various datasets, achieving decreases of 5.3%, 5.7%, 1.2%, 13.4%, 4.1%, 2.9%, 3.1%, and 3.8% for the Weather, Traffic, Electricity, ILI, ETTh1, ETTh2, ETTm1, and ETTm2 datasets, respectively. On average, our approach improved the MAE metric by 4.9% across all datasets, signifying its consistent effectiveness.

In conclusion, our proposed method not only showcases superior performance compared to other approaches but also demonstrates its robustness in various settings, making it a compelling choice for multivariate time series prediction tasks.

MSLSTM ablation study To demonstrate the effectiveness of the Multi-scale LSTM, we conducted two experiments on five datasets (ILI, ETTh1, ETTh2, ETTm1,

and ETTm2): comparing the performance of Single-scale and Multi-scale LSTM; and comparing the performance of the Multi-scale LSTM with the Multi-scale Transformer, where we adjusted the Transformer parameters (d_model and d_ff) by doubling them(Transformer*) and used the same setting as the PatchTST (Nie et al. 2023). Table 3 demonstrates the efficacy of the Multi-scale approach, enabling the capture of a greater array of temporal features within time series data. This facilitates the model's acquisition of distinct features at varying resolutions. In comparison with the Multi-scale Transformer, our findings indicate the superiority of employing LSTM as the underlying network. Additionally, it is observed that as model complexity increases, Transformer is more susceptible to overfitting.

Methods		MSLSTM		Single-scale LSTM		MSTransformer		MSTransformer*	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ILI	24	**1.298**	**0.665**	1.341	0.678	1.392	0.695	1.359	0.685
	36	**1.241**	**0.676**	1.273	0.688	1.349	0.718	1.413	0.729
	48	1.530	0.750	**1.504**	**0.749**	1.582	0.772	1.598	0.773
	60	**1.406**	**0.731**	1.444	0.736	1.460	0.753	1.453	0.755
ETTh1	96	0.360	**0.386**	0.375	0.395	0.360	**0.386**	**0.357**	**0.386**
	192	0.402	**0.412**	0.418	0.419	0.407	0.417	**0.401**	0.413
	336	**0.414**	**0.421**	0.420	0.424	0.425	0.427	0.424	0.427
	720	**0.420**	**0.446**	0.429	0.450	**0.420**	**0.446**	0.423	0.449
ETTh2	96	**0.274**	**0.328**	0.294	0.342	**0.274**	0.329	0.276	0.329
	192	**0.339**	**0.371**	0.357	0.384	0.344	0.374	0.344	0.373
	336	**0.330**	**0.373**	0.335	0.384	0.336	0.378	0.341	0.379
	720	**0.382**	**0.415**	0.403	0.432	0.386	0.422	0.385	0.420
ETTm1	96	0.283	0.328	0.289	0.329	**0.280**	**0.325**	0.291	0.338
	192	**0.324**	**0.355**	0.334	0.356	0.331	0.356	0.342	0.368
	336	**0.356**	**0.376**	0.368	0.378	0.365	0.377	0.378	0.389
	720	0.424	0.420	0.424	0.414	**0.419**	**0.412**	0.429	0.420
ETTm2	96	0.163	0.246	0.162	0.246	**0.160**	**0.243**	0.162	0.248
	192	**0.216**	**0.283**	**0.216**	0.284	**0.216**	**0.283**	0.222	0.287
	336	**0.264**	**0.315**	0.266	0.317	**0.264**	0.316	0.277	0.327
	720	**0.348**	**0.370**	0.351	**0.370**	0.350	0.372	0.367	0.383

- MS=Multi-Scale; Transformer* refers to a Transformer with a larger number of parameters.

Table 3: The performance comparisons of single-scale, multi-scale, LSTM and Transformer . The best results are in **bold** . The second best results are in underlined .

Loss Functions		SQF		SQF-PCT		SQF-MAE		SQF-RQF		MSE	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ILI	24	1.298	**0.665**	1.299	**0.665**	1.246	0.672	1.298	0.666	**1.228**	0.668
	36	1.241	**0.676**	**1.239**	**0.676**	1.320	0.743	1.242	0.678	1.346	0.748
	48	1.530	0.750	**1.528**	**0.749**	1.549	0.809	1.530	**0.749**	1.573	0.814
	60	1.406	**0.731**	1.406	**0.731**	1.525	0.829	**1.404**	**0.731**	1.630	0.849
ETTh1	96	**0.360**	**0.386**	0.366	0.387	0.365	0.385	**0.360**	0.387	0.375	0.398
	192	**0.402**	**0.412**	0.414	0.413	0.413	**0.412**	0.403	0.413	0.417	0.422
	336	**0.414**	**0.421**	0.432	0.424	0.426	0.423	0.416	0.426	0.422	0.431
	720	0.420	0.446	0.435	0.449	**0.412**	**0.439**	0.446	0.466	0.484	0.486
ETTh2	96	**0.274**	0.328	0.275	0.328	0.276	**0.327**	**0.274**	0.329	0.281	0.337
	192	0.339	0.371	0.340	0.371	0.341	**0.370**	**0.338**	0.372	0.355	0.387
	336	**0.330**	**0.373**	0.335	0.376	0.332	**0.373**	0.331	0.374	0.348	0.393
	720	0.382	0.415	0.386	0.416	**0.378**	**0.406**	0.383	0.417	0.396	0.432
ETTm1	96	**0.283**	**0.328**	0.289	**0.328**	0.450	0.362	0.284	0.330	0.293	0.345
	192	**0.324**	**0.355**	0.328	**0.355**	0.517	0.393	**0.324**	0.356	0.333	0.371
	336	**0.356**	**0.376**	0.364	0.377	0.558	0.414	0.357	0.377	0.363	0.389
	720	0.424	0.420	0.428	**0.412**	0.587	0.442	**0.421**	0.422	0.424	0.424
ETTm2	96	0.163	0.246	0.163	**0.244**	0.167	0.247	**0.162**	0.246	0.164	0.253
	192	0.216	**0.283**	0.220	0.284	0.222	0.285	**0.215**	**0.283**	0.222	0.292
	336	**0.264**	**0.315**	0.269	0.317	0.268	0.316	**0.264**	**0.315**	0.270	0.324
	720	0.348	**0.370**	0.353	0.371	0.353	0.371	**0.347**	**0.370**	0.357	0.381

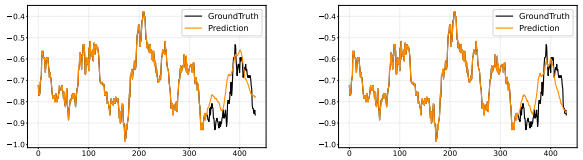
Table 4: The ablation study on the Smooth Quadratic Function (SQF) using the MSLSTM model .

Smooth quadratic function To validate the effectiveness of each term in the SQF, we conducted ablation experiments by removing the Penalty constraint term (PCT), MAE, and rational quadratic function (RQF) and tested the multivariate prediction performance on five datasets (ETTh1, ETTh2, ETTm1, ETTm2, and ILI). Table 4 shows that each term in SQF has its effectiveness. Additionally, we applied SQF to both PatchTST/42 (Nie et al. 2023) and Informer (Zhou et al. 2021) models, and Table 5 demonstrates the results of multivariate prediction on three datasets. In most cases, the model with SQF outperformed the one with MSE training metric. This result validates the capability of SQF in improving the model's learning performance.

Methods		PatchTST/42+MSE		PatchTST/42+SQF		Informer+MSE		Informer+SQF	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ILI	24	1.522	0.814	**1.412**	**0.722**	4.657	1.449	**4.603**	**1.406**
	36	1.430	0.834	**1.351**	**0.714**	4.650	1.463	**4.614**	**1.430**
	48	1.673	0.854	**1.634**	**0.789**	**5.004**	**1.542**	5.402	1.561
	60	1.529	0.862	**1.478**	**0.750**	**5.071**	**1.543**	5.272	1.564
ETTh1	96	0.375	0.399	**0.360**	**0.387**	0.941	0.769	**0.846**	**0.680**
	192	0.414	0.421	**0.407**	**0.418**	1.007	0.786	**0.976**	**0.723**
	336	0.431	0.436	**0.422**	**0.426**	1.038	0.784	**1.027**	**0.762**
	720	0.449	0.466	**0.427**	**0.452**	1.144	0.857	**1.136**	**0.817**
ETTh2	96	0.274	0.336	**0.273**	**0.329**	**1.549**	**0.952**	1.779	1.004
	192	**0.339**	0.379	0.341	**0.373**	3.792	1.542	**1.878**	**1.042**
	336	**0.331**	0.380	**0.331**	**0.376**	4.215	1.642	**2.115**	**1.097**
	720	0.379	0.422	**0.378**	**0.415**	3.656	1.619	**2.057**	**1.111**
ETTm1	96	0.290	0.342	**0.287**	**0.329**	0.626	0.560	**0.524**	**0.482**
	192	0.332	0.369	**0.331**	**0.357**	0.725	0.619	**0.692**	**0.565**
	336	**0.366**	0.392	0.371	**0.380**	1.005	0.741	**0.818**	**0.647**
	720	**0.420**	0.424	0.424	**0.414**	1.133	0.845	**1.108**	**0.800**
ETTm2	96	0.165	0.255	**0.161**	**0.246**	0.355	0.462	**0.332**	**0.402**
	192	0.220	0.292	**0.219**	**0.286**	**0.595**	**0.586**	0.816	0.654
	336	0.278	0.329	**0.269**	**0.320**	1.270	0.871	**1.176**	**0.844**
	720	0.367	0.385	**0.348**	**0.371**	3.001	1.267	**2.224**	**1.141**

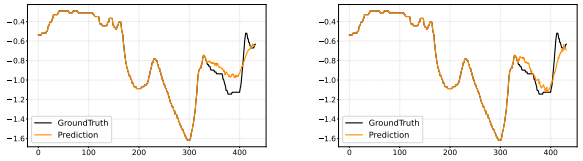
Table 5: Experimental results on five real-world datasets with PatchTST/42 and Informer.

Furthermore, we analyze the prediction curves. Figure 4 depicts the results of predicting the next 96 steps on the ETTm1 dataset using MSE and SQF, while Figure 5 demonstrates the outcomes of predicting the next 96 steps on the ETTm2 dataset employing MSE and SQF. From the figures, it can be noticed that the prediction results utilizing the SQF are more robust, and the prediction curves are smoother.



(a) Results predicted using MSE (b) Results predicted using SQF

Figure 4: Predicting curves on ETTm1



(a) Results predicted using MSE (b) Results predicted using SQF

Figure 5: Predicting curves on ETTm2

In summary, we designed experiments from multiple perspectives to verify the superiority of the SQF and found that it can adapt to various datasets and models. It enables the model to capture high-dimensional information in time-series data.

6 Conclusion and Discussion

This paper introduces a novel method for multivariate time series forecasting. To effectively capture multi-scale information inherent in time series data, we present the Multi-scale approach, which extracts crucial details of the underlying dynamics and interactions within the time series. Consequently, we develop the Multi-scale LSTM (MSLSTM) model, specifically tailored for accurate multivariate time series prediction. To further enhance model training and improve noise resistance, we introduce the Smooth Rational Function (SQF), inspired by kernel methods. The integration of our proposed model and loss function leads to remarkable performance, setting a new state-of-the-art on diverse real-world datasets.

References

Bai, S.; Kolter, J. Z.; and Koltun, V. 2018. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*.

Bao, Y.; and Ullah, A. 2007. The second-order bias and mean squared error of estimators in time-series models. *Journal of Econometrics*, 140(2): 650–669.

Bartholomew, D. J. 1971. Time Series Analysis Forecasting and Control.

Chen, T.; He, T.; Benesty, M.; Khotilovich, V.; Tang, Y.; Cho, H.; Chen, K.; Mitchell, R.; Cano, I.; Zhou, T.; et al. 2015. Xgboost: extreme gradient boosting. *R package version 0.4-2*, 1(4): 1–4.

Cuturi, M.; and Blondel, M. 2017. Soft-dtw: a differentiable loss function for time-series. In *International conference on machine learning*, 894–903. PMLR.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Dey, R.; and Salem, F. M. 2017. Gate-variants of gated recurrent unit (GRU) neural networks. In *2017 IEEE 60th international midwest symposium on circuits and systems (MWSCAS)*, 1597–1600. IEEE.

Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

Graves, A.; and Graves, A. 2012. Long short-term memory. *Supervised sequence labelling with recurrent neural networks*, 37–45.

Han, L.; Ye, H.-J.; and Zhan, D.-C. 2023. The Capacity and Robustness Trade-off: Revisiting the Channel Independent Strategy for Multivariate Time Series Forecasting. *arXiv preprint arXiv:2304.05206*.

Hearst, M. A.; Dumais, S. T.; Osuna, E.; Platt, J.; and Scholkopf, B. 1998. Support vector machines. *IEEE Intelligent Systems and their applications*, 13(4): 18–28.

Kadiyala, A.; and Kumar, A. 2014. Multivariate time series models for prediction of air quality inside a public transportation bus using available software. *Environmental Progress & Sustainable Energy*, 33(2): 337–341.

Kardakos, E. G.; Alexiadis, M. C.; Vagropoulos, S. I.; Simoglou, C. K.; Biskas, P. N.; and Bakirtzis, A. G. 2013. Application of time series and artificial neural network models in short-term forecasting of PV power generation. In *2013 48th International Universities' Power Engineering Conference (UPEC)*, 1–6. IEEE.

Karevan, Z.; and Suykens, J. A. 2020. Transductive LSTM for time-series prediction: An application to weather forecasting. *Neural Networks*, 125: 1–9.

Lai, G.; Chang, W.-C.; Yang, Y.; and Liu, H. 2018. Modeling long-and short-term temporal patterns with deep neural networks. In *The 41st international ACM SIGIR conference on research & development in information retrieval*, 95–104.

Le Guen, V.; and Thome, N. 2019. Shape and time distortion loss for training deep time series forecasting models. *Advances in neural information processing systems*, 32.

Li, S.; Jin, X.; Xuan, Y.; Zhou, X.; Chen, W.; Wang, Y.-X.; and Yan, X. 2019. Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. *Advances in neural information processing systems*, 32.

Liu, S.; Yu, H.; Liao, C.; Li, J.; Lin, W.; Liu, A. X.; and Dustdar, S. 2021. Pyraformer: Low-complexity pyramidal attention for long-range time series modeling and forecasting. In *International conference on learning representations*.

Maclaurin, C. 1742. *A treatise of fluxions: in two books*, volume 1. Ruddimans.

Medsker, L. R.; and Jain, L. 2001. Recurrent neural networks. *Design and Applications*, 5(64-67): 2.

Merity, S.; McCann, B.; and Socher, R. 2017. Revisiting activation regularization for language rnns. *arXiv preprint arXiv:1708.01009*.

Morid, M. A.; Sheng, O. R. L.; and Dunbar, J. 2023. Time series prediction using deep learning methods in healthcare. *ACM Transactions on Management Information Systems*, 14(1): 1–29.

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

Radford, A.; Narasimhan, K.; Salimans, T.; Sutskever, I.; et al. 2018. Improving language understanding by generative pre-training.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Wang, H.; Peng, J.; Huang, F.; Wang, J.; Chen, J.; and Xiao, Y. 2023. MICN: Multi-scale Local and Global Context Modeling for Long-term Series Forecasting.

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

Wu, H.; Xu, J.; Wang, J.; and Long, M. 2021. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. *Advances in Neural Information Processing Systems*, 34: 22419–22430.

Zeng, A.; Chen, M.; Zhang, L.; and Xu, Q. 2023. Are transformers effective for time series forecasting? In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, 11121–11128.

Zhou, H.; Zhang, S.; Peng, J.; Zhang, S.; Li, J.; Xiong, H.; and Zhang, W. 2021. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, 11106–11115.

Zhou, T.; Ma, Z.; Wen, Q.; Wang, X.; Sun, L.; and Jin, R. 2022. Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *International Conference on Machine Learning*, 27268–27286. PMLR.