Rankformer: Leveraging Rank Correlation for Transformer-based Time Series Forecasting

Zuokun Ouyang, Meryem Jabloun, Philippe Ravier

Rankformer: Leveraging Rank Correlation for Transformer-based Time Series Forecasting

Zuokun Ouyang, Meryem Jabloun, Philippe Ravier
PRISME Laboratory, University of Orléans, France
{zuokun.ouyang,meryem.jabloun,philippe.ravier}@univ-orleans.fr

***Abstract*—Long-term forecasting problem for time series has been actively studied during the last several years, and preceding Transformer-based models have exploited various self-attention mechanisms to discover the long-range dependencies. However, the hidden dependencies required by the forecasting task are not always appropriately extracted, especially the nonlinear serial dependencies in some datasets. In this paper, we propose a novel Transformer-based model, namely Rankformer, leveraging the rank correlation function and decomposition architecture for long-term time series forecasting tasks. Rankformer outperforms four state-of-the-art Transformer-based models and two RNN-based models for different forecasting horizons on different datasets on which extensive experiments were conducted.**

***Index Terms*—Time Series, Forecasting, Transformer, Rank Correlation, Nonlinear Dependencies**

I. Introduction

Time series forecasting is a process for predicting the future values of a given time series based on its historical behavior by developing a model describing its underlying characteristics and extrapolating into the future. It has been widely used in many applications, such as weather forecasting [1], GDP prediction [2], traffic forecasting [3], and energy consumption prediction [4]. It has been dominated for a few decades by statistical methods such as AutoRegression Integrated Moving Average (ARIMA), ExponenTial Smoothing (ETS), and Theta method [5]–[8]. In the past few years, deep learning has been applied to time series forecasting and achieved great success [9]–[12]. The most popular deep learning models for time series forecasting include Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and Transformer models. CNNs and RNNs have been widely exploited in forecasting tasks due to their ability to capture sequential/temporal dependencies in the time series [13]. Some representative works include LSTNet [14], DeepAR [15], and TCN [16].

Since its first birth in 2017, Transformer models have become increasingly popular and applied successfully in various fields, including machine translation, computer vision, and text generation, to list a few [17]–[20]. In the time series domain, Informer [21] was the first work that introduced Transformer for time series forecasting with a ProbSparse self-attention calculation and a self-attention distilling mechanism to handle the quadratic computational complexity. Autoformer [22] substitutes the self-attention block with an AutoCorrelation mechanism to discover the period-based dependencies and adopts a decomposition structure to separate the long-term stationary trend and the seasonal patterns. Other Transformer models were also applied to time series forecasting tasks, such as Reformer [23], which employed locally sensitive hashing self-attention, and LogTrans [24], which uses a heuristic method to reduce the complexity of the self-attention mechanism.

Nevertheless, the formerly mentioned Transformer models have not been able to exploit the long-range dependencies in time series fully, especially the nonlinear serial dependencies. Informer applied the ProbSparse self-attention mechanism to reduce the computational complexity, but the hidden long-range dependency was not extracted properly. Autoformer used the AutoCorrelation mechanism to discover the period-based dependencies. However, the AutoCorrelation used in the model is based on the Pearson correlation function, which supports only linear correlation even on an NN basis, while in some time series, the long-term dependencies are nonlinear.

In this paper, we propose Rankformer, a novel Transformer model for long-term forecasting tasks, leveraging the rank correlation function and a decomposition architecture for time series forecasting tasks. Rankformer outperforms other Transformer-based models in extensive experiments on four forecasting benchmark datasets for four forecasting horizons.

The rest of the paper is organized as follows. Sec. II introduces the proposed Rankformer model. We then present the experimental setups and configurations in Sec. III. The comparison and discussions based on the results are given in Sec. IV. Finally, Sec. V concludes the paper.

II. Methods

In this section, we introduce the architecture of Rankformer and its two key components, i.e., the Rank Correlation and Multi-Level Decomposition modules.

A. Rankformer Architecture

As shown in Fig. 1, Rankformer has an encoder-decoder architecture per Autoformer [22]. The encoder is composed of a stack of N identical layers, each containing one multi-head Rank Correlation (*RankCorr*) block, two Multi-Level Decomposition (*MLDecomp*) blocks, and one Feed-Forward (*FF*) block. The decoder is a stack of M identical layers, each of which is composed of two multi-head RankCorr blocks, three MLDecomp blocks, and one FF block. Combining the outputs of the last MLDecomp block and the refined trend-cyclical part in the decoder composes the final prediction. The architecture of Rankformer is detailed in the following contents.

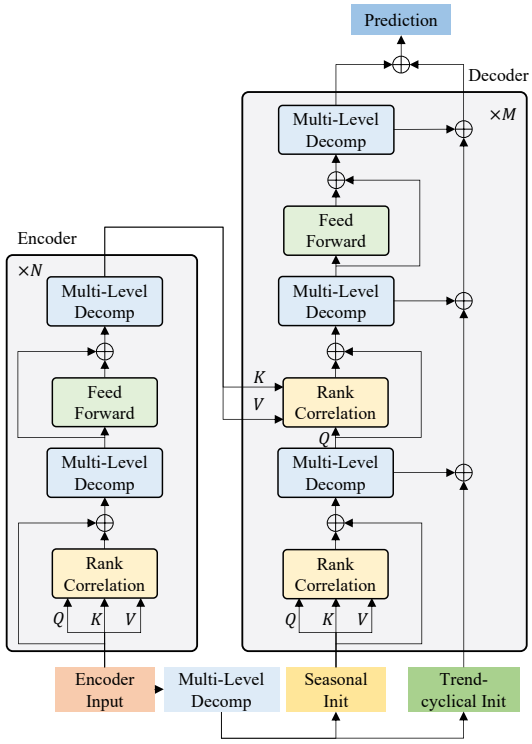

Fig. 1. The architecture of Rankformer.

1) Encoder: With the RankCorr and MLDecomp blocks, the encoder decomposes the series into seasonal and trend-cyclical parts. With the latter being neglected during the modeling process, the encoder mainly models the seasonal component. The output of the l-th encoder layer can be summarized as $\mathcal{X}_{\text{en}}^{l} = \text{Encoder}(\mathcal{X}_{\text{en}}^{l-1})$ and the process in one encoder layer is expressed as follows:

$$\begin{aligned} \mathcal{S}_{\text{en}}^{l,1}, \_ &= \text{MLDecomp}\big(\text{RankCorr}(\mathcal{X}_{\text{en}}^{l-1}) + \mathcal{X}_{\text{en}}^{l-1}\big), \\ \mathcal{S}_{\text{en}}^{l,2}, \_ &= \text{MLDecomp}\big(\text{FF}(\mathcal{S}_{\text{en}}^{l,1}) + \mathcal{S}_{\text{en}}^{l,1}\big), \end{aligned} \tag{1}$$

where $\mathcal{S}_{\text{en}}^{l,i}$ denotes the seasonal component after the i-th MLDecomp block and $\mathcal{X}_{\text{en}}^{l} = \mathcal{S}_{\text{en}}^{l,2},\ l \in \{1, 2, ..., N\}$.

2) Decoder: The decoder in Rankformer has two streams, i.e., the trend-cyclical stream and the seasonal stream. While the seasonal stream continuously refines the seasonal part of the time series, the trend-cyclical stream focuses on modeling the trend-cyclical component. With a similar notation as per encoder, we can summarize the process in one decoder layer as $\mathcal{X}_{\text{de}}^{l}, \mathcal{T}_{\text{de}}^{l} = \text{Decoder}(\mathcal{X}_{\text{de}}^{l-1}, \mathcal{T}_{\text{de}}^{l-1})$ and formalize it as follows:

$$\begin{aligned} \mathcal{S}_{\text{de}}^{l,1}, \mathcal{T}_{\text{de}}^{l,1} &= \text{MLDecomp}\Big(\text{RankCorr}(\mathcal{X}_{\text{de}}^{l-1}) + \mathcal{X}_{\text{de}}^{l-1}\Big), \\ \mathcal{S}_{\text{de}}^{l,2}, \mathcal{T}_{\text{de}}^{l,2} &= \text{MLDecomp}\Big(\text{RankCorr}(\mathcal{S}_{\text{de}}^{l,1}, \mathcal{X}_{\text{en}}^{N}) + \mathcal{S}_{\text{de}}^{l,1}\Big), \\ \mathcal{S}_{\text{de}}^{l,3}, \mathcal{T}_{\text{de}}^{l,3} &= \text{MLDecomp}\Big(\text{FF}(\mathcal{S}_{\text{de}}^{l,2}) + \mathcal{S}_{\text{de}}^{l,2}\Big), \\ \mathcal{T}_{\text{de}}^{l} &= \mathcal{T}_{\text{de}}^{l-1} + W_{l,1}\mathcal{T}_{\text{de}}^{l,1} + W_{l,2}\mathcal{T}_{\text{de}}^{l,2} + W_{l,3}\mathcal{T}_{\text{de}}^{l,3}, \end{aligned} \tag{2}$$

where $\mathcal{S}_{\text{en}}^{l,i}$ and $\mathcal{T}_{\text{de}}^{l,i}$ are the seasonal and trend-cyclical components respectively, and $W_{l,1}$, $W_{l,2}$, $W_{l,3}$ are trainable weights. The outputs of the l-th decoder layer are two fold: the refined seasonal patterns $\mathcal{X}_{\text{de}}^{l} = \mathcal{S}_{\text{de}}^{l,3}$, and the multiple level trend-cyclical patterns $\mathcal{T}_{\text{de}}^{l}$, where $l \in \{1, 2, ..., M\}$.

3) Model Inputs and Outputs: We denote the input length as I, the output length as O, and the model dimension as d. There are three inputs for Rankformer:

- The encoder input are the last I time steps in the time series: $\mathcal{X}_{\text{en}} \in \mathbb{R}^{I \times d}$.
- The seasonal stream input concatenates the latter half of the encoder's decomposed input and a length-O placeholder with zeros: $\mathcal{X}_{\text{de,S}} = \text{concat}(\mathcal{X}_{\text{en,S}}, \mathcal{X}_0) \in \mathbb{R}^{(\frac{I}{2}+O) \times d}$.
- The trend-cyclical stream input also consists of the latter half of the decomposed $\mathcal{X}_{\text{en}}$ and a placeholder filled by the average of $\mathcal{X}_{\text{en}}$: $\mathcal{X}_{\text{de,T}} = \text{concat}(\mathcal{X}_{\text{en,T}}, \mathcal{X}_{\text{avg}}) \in \mathbb{R}^{(\frac{I}{2}+O) \times d}$.

The relationship between the inputs can be formalized as follows:

$$\begin{aligned} \mathcal{X}_{\text{en,S}}, \mathcal{X}_{\text{en,T}} &= \text{MLDecomp}\left(\mathcal{X}_{\text{en}}\left[\frac{I}{2} : I\right]\right), \\ \mathcal{X}_{\text{de,S}} &= \text{concat}(\mathcal{X}_{\text{en,S}}, \mathcal{X}_0), \\ \mathcal{X}_{\text{de,T}} &= \text{concat}(\mathcal{X}_{\text{en,T}}, \mathcal{X}_{\text{avg}}). \end{aligned} \tag{3}$$

The final output of the model is a combination of the seasonal and the trend-cyclical streams in the decoder: $W_{\text{S}}\mathcal{X}_{\text{de}}^{M} + \mathcal{T}_{\text{de}}^{M}$, where W_{S} is a trainable weight to project $\mathcal{X}_{\text{de}}^{M}$ into the target dimension.

B. RankCorrelation Block

The Pearson correlation coefficient, also known as Pearson's ρ, is widely used to measure the linear correlation between two variables. Given two random variables X and Y, Pearson's ρ defined as follows:

$$\rho_p(X, Y) = \frac{\text{cov}(X, Y)}{\sigma(X)\sigma(Y)}. \tag{4}$$

The AutoCorrelation Function (ACF) adopts the Pearson correlation function to measure the correlation between two distant time points in a stationary time series y_t:

$$\text{ACF}(k) = \rho_p(y_{t-k}, y_t) = \frac{\text{cov}(y_{t-k}, y_t)}{\sigma(y_{t-k})\sigma(y_t)},\ k = 0, 1, 2, ..., \forall t. \tag{5}$$

However, in some time series, the long-term dependencies are not linear. In this case, the nonlinearity can cause a low Pearson's ρ and thus result in an erroneous dependencies measurement. To address this issue, we propose to use the Rank Correlation Function (*RCF*), more generally known as Spearman's ρ [25], to measure the nonlinear correlation. Spearman's ρ is defined as follows:

$$\rho_s(X, Y) = \frac{\text{cov}(R(X), R(Y))}{\sigma(R(X))\sigma(R(Y))}, \tag{6}$$

where $R(X)$ and $R(Y)$ are the ranks of X and Y. ρ denotes the usual Pearson correlation coefficient but is applied to the rank variables, which is leveraged to compute the Ranked ACF (*RACF*). ρ_s is defined in $[-1, 1]$, where -1 indicates a perfect

negative monotonic relationship, 0 indicates no monotonic relationship, and 1 indicates a perfect positive monotonic relationship. ρ_s is invariant to monotonic transformations of the variables and is robust to outliers. Therefore, it is more suitable for stationary time series with nonlinear serial dependencies. Our RACF is defined as:

$$\mathrm{RACF}(k) = \rho_s(y_{t-k}, y_t),\ k = 0, 1, 2, ..., \forall t. \tag{7}$$

In our implementation, the RACF is computed by exploiting the FFT, which accelerates the Fourier transform to $\mathcal{O}(N \log N)$, and the Wiener-Khinchin theorem, which states that the ACF of a stationary time series can be computed by the Fourier transform of its power spectrum. The ranking procedure is supported by the `torchsort`[1] library, which offers an efficient $\mathcal{O}(N \log N)$ sorting operator [26]. Thus, the RACF is computed by ranking the time series and then computing the ACF of the ranked time series. The total computational complexity of calculating the RACF is thus $\mathcal{O}(N \log N)$.

C. Multi-Level Decomposition Block

We adopted a multi-level decomposition block to decompose the input time series into the seasonal and trend-cyclical components. The block consists of Multiple Moving Average (*MMA*) filters with varying kernel sizes to yield different trend-cyclical components, rather than just one fix-length MA filter in Autoformer. The MLDecomp block is formalized as:

$$\mathcal{X}_{\text{trend-cyclical}} = \sum_{k=1}^{K} W_{\text{decomp},k} \cdot \mathrm{MMA}(\mathcal{X}_{\text{input}}, k), \tag{8}$$

where K is a set of kernel sizes, $W_{\text{decomp},k}$ is a trainable weight tensor, and MMA are multiple moving average filters. The output of the MLDecomp block is a weighted sum of the trend-cyclical components.

III. Experiments

This section presents our experimental settings and results.

A. Datasets

Rankformer was tested with other state-of-the-art methods on four well-known datasets:

- Electricity Transformer Temperature (ETT) [21]: Oil temperature and six power load features recorded every 15 minutes from July 2016 to July 2018 in two Chinese counties.
- Exchange-Rate[2]: Daily exchange rates of eight countries, i.e., Australia, British, Canada, China, Japan, New Zealand, Singapore, and Switzerland, from 1990 to 2016.
- Weather[3]: 10-minute level local climate data containing 21 meteorological features for 2020 collected by Max-Planck-Institut für Biogeochemie, Jena.

[1]https://github.com/teddykoker/torchsort

[2]https://github.com/laiguokun/multivariate-time-series-data

[3]https://www.bgc-jena.mpg.de/wetter/

TABLE I
Dataset Description

Dataset	ETT	Exchange	Weather	ILI
Length	69680	7588	52696	966
Dimension	7	8	21	7
Frequency	15 min	1 day	10 min	1 week
Engle's test p-value	$<$ 2.2e-16	$<$ 2.2e-16	$<$ 2.2e-16	0.8126
ARCH effect	Significant	Significant	Significant	Insignificant

- Influenza-Like Illness (ILI)[4]: Weekly ILI patients data from the U.S. Centers for Disease Control and Prevention between 2002 to 2021, containing the ratio of ILI patients and the total number of patients.

All datasets were separated into train/validation/test sets in chronological order with a 7/1/2 split, except for the ETT dataset, which was split into 6/2/2, as per Autoformer [22] and Informer [21]. The datasets' statistics are listed in the first four rows of Tab. I.

We also evaluated the significance of the nonlinearity in the serial dependencies by performing Engle's Lagrange Multiplier Test [27] on the four datasets. It assesses the significance of autoregressive conditional heteroskedasticity (ARCH) effects in a time series. A significant result reveals nonlinear serial dependencies in the series. The test results are listed in the last two rows of Tab. I.

B. Experimental Settings

We kept the same number of encoder-decoder layers as Autoformer: two encoder layers and one decoder layer. Rankformer was trained using the Mean Square Error loss and the Adam optimizer [28] with an initial learning rate of 10^{-4}. The batch size was set to 32. The model was trained for ten epochs with a learning rate scheduler that reduces the learning rate by a factor of 0.5 when the validation loss plateaus. The model was implemented in PyTorch [29] and trained on a single NVIDIA Tesla V100 GPU.

IV. Results and Discussion

We compared Rankformer with the following state-of-the-art methods: Autoformer [22], Informer [21], LogTrans [24], Reformer [23], LSTNet [14], and LSTM [30]. We used the Mean Square Error (MSE) and the Mean Absolute Error (MAE) as the evaluation metrics, and we fixed the input length to 36 for ILI and 96 for others as per Autoformer. The results are presented in Tab. II. The best results are highlighted in bold, and the second-best results are highlighted with underscores.

Overall, Rankformer outperforms the other methods on the ETT, Exchange-Rage, and Weather datasets and is slightly weaker than Autoformer on the ILI dataset. Particularly, under the Input-96-Output-96 setting, Rankformer yields **13.3%**

[4]https://gis.cdc.gov/grasp/fluview/fluportaldashboard.html

TABLE II
FORECASTING RESULTS FOR DIFFERENT MODELS ON DIFFERENT FORECAST HORIZONS

Models		Rankformer		Autoformer [22]		Informer [21]		LogTrans [24]		Reformer [23]		LSTNet [14]		LSTM [30]	
Metric		MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
ETT	24	**0.149**	**0.258**	0.153	0.261	0.173	0.301	0.211	0.332	0.333	0.429	1.101	0.831	0.580	0.572
	48	0.183	0.281	**0.178**	**0.280**	0.303	0.409	0.427	0.487	0.558	0.571	2.619	1.393	0.747	0.630
	96	**0.221**	**0.302**	0.255	0.339	0.365	0.453	0.768	0.642	0.658	0.619	3.142	1.365	2.041	1.073
	288	**0.314**	**0.357**	0.342	0.378	1.047	0.804	1.090	0.806	2.441	1.190	2.856	1.329	0.969	0.742
Exchange	96	**0.162**	**0.290**	0.197	0.323	0.847	0.752	0.968	0.812	1.065	0.829	1.551	1.058	1.453	1.049
	192	**0.251**	**0.365**	0.300	0.369	1.204	0.895	1.040	0.851	1.188	0.906	1.477	1.028	1.846	1.179
	336	**0.428**	**0.486**	0.509	0.524	1.672	1.036	1.659	1.081	1.357	0.976	1.507	1.031	2.136	1.231
	720	**1.157**	**0.837**	1.447	0.941	2.478	1.310	1.941	1.127	1.510	1.016	2.285	1.243	2.984	1.427
Weather	96	**0.263**	**0.332**	0.266	0.336	0.300	0.384	0.458	0.490	0.689	0.596	0.594	0.587	0.369	0.406
	192	**0.298**	**0.356**	0.307	0.367	0.598	0.544	0.658	0.589	0.752	0.638	0.560	0.565	0.416	0.435
	336	**0.350**	**0.390**	0.359	0.395	0.578	0.523	0.797	0.652	0.639	0.596	0.597	0.587	0.455	0.454
	720	0.430	0.435	**0.419**	**0.428**	1.059	0.741	0.869	0.675	1.130	0.792	0.618	0.599	0.535	0.520
ILI	24	3.556	1.319	**3.483**	**1.287**	5.764	1.677	4.480	1.444	4.400	1.382	6.026	1.770	5.914	1.734
	36	**2.821**	**1.112**	3.103	1.148	4.755	1.467	4.799	1.467	4.783	1.448	5.340	1.668	6.631	1.845
	48	2.907	1.144	**2.669**	**1.085**	4.763	1.469	4.800	1.468	4.832	1.465	6.080	1.787	6.736	1.857
	60	3.232	1.239	**2.770**	**1.125**	5.264	1.564	5.278	1.560	4.882	1.483	5.548	1.720	6.870	1.879

MSE reduction on ETT and **17.5%** on Exchange-Rate, compared to Autoformer. It also outperforms Autoformer on the Weather dataset, but the difference is not significant.

The results on the Exchange-Rate dataset are particularly impressive. Despite the fact that Exchange-Rate is a very challenging dataset without any notable periodicity, Rankformer still gives the best improvement over Autoformer. We attribute this to the nonlinear serial dependencies in the dataset being captured more properly by Rankformer than by Autoformer.

On the contrary, due to the high linear correlation in the ILI dataset, Rankformer is not able to outperform Autoformer. In fact, the p-value of Engle's Lagrange Multiplier test of the ILI dataset is 0.8126 ($\gg 0.05$), which means that there are statistically significant linear serial dependencies inside the ILI series that can be handled more appropriately by Autoformer.

On the other hand, thanks to the FFT and Wiener-Khinchin theorem, Rankformer achieves an $\mathcal{O}(N \log N)$ complexity. It is not only a huge advantage in the computing speed compared to the original Transformer's $\mathcal{O}(N^2)$ complexity but also brings the convenience of nonlinear serial dependencies measurement to Autoformer without augmenting the time complexity. This makes Rankformer much more efficient than the Transformer, especially when the input sequence is long, and also more appropriate for forecasting time series with nonlinear serial dependencies.

Rankformer and Autoformer are very similar in terms of their measurement of correlation. The only difference is that Rankformer uses a rank-based ACF, i.e., the RACF, while Autoformer uses a value-based ACF, which means, with an optimized sorting and ranking operator, the RACF can be easily integrated into Autoformer and enables it for nonlinear-time-dependencies-measurement. Nonetheless, a small difference in the dependencies measurement can leverage a decent improvement in the forecasting performance, especially for the datasets with nonlinear serial dependencies.

V. CONCLUSION

In this paper, we propose a novel method, Rankformer, for forecasting time series, especially those with nonlinear serial dependencies. Rankformer is based on the Transformer architecture and uses a rank-based ACF to measure the nonlinear serial dependencies. We show that Rankformer outperforms Autoformer, a state-of-the-art method with linear serial dependencies measurement, on three real-world datasets. We also show that Rankformer is more efficient than the original Transformer in terms of both computing speed and memory usage. Furthermore, we show that the RACF used by Rankformer can be easily integrated into Autoformer to improve its performance on datasets with nonlinear serial dependencies. In fact, in most nonlinear serial dependencies cases, the series shows an ARCH effect, which can be captured by the RACF. Therefore, we believe that Rankformer can be a good alternative to Autoformer for forecasting time series with nonlinear serial dependencies. In the future, we plan to investigate: 1) the robustness of Rankformer in the presence of perturbation; 2) the possibility of integrating other nonlinear serial dependencies measurement methods and the general ARCH (GARCH) characteristics to further improve the performance of Rankformer.

ACKNOWLEDGMENT

This research was partly supported by Association Nationale de la Recherche et de la Technologie (CIFRE 2019/0551).

REFERENCES

[1] X. Shi *et al.*, "Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting," in *Proc. NeurIPS*, 2015.

[2] L. Longo, M. Riccaboni, and A. Rungi, "A neural network ensemble approach for GDP forecasting," *J. Econ. Dyn. Control*, vol. 134, p. 104 278, 2022.

[3] J. Zuo *et al.*, "Graph convolutional networks for traffic forecasting with missing values," *Data. Min. Knowl. Disc.*, vol. 37, no. 2, pp. 913–947, 2023.

[4] Y. Yaslan and B. Bican, "Empirical mode decomposition based denoising method with support vector regression for time series prediction: A case study for electricity load forecasting," *Measurement*, vol. 103, pp. 52–61, 2017.

[5] G. E. P. Box *et al.*, *Time Series Analysis: Forecasting and Control*, 5th ed. John Wiley & Sons, Inc., 2015.

[6] R. J. Hyndman and G. Athanasopoulos, *Forecasting: Principles and Practice*, 3rd ed. OTexts, 2021.

[7] V. Assimakopoulos and K. Nikolopoulos, "The theta model: A decomposition approach to forecasting," *Int. J. Forecast.*, vol. 16, no. 4, pp. 521–530, 2000.

[8] Z. Ouyang, P. Ravier, and M. Jabloun, "STL Decomposition of Time Series Can Benefit Forecasting Done by Statistical Methods but Not by Machine Learning Ones," *Eng. Proc.*, vol. 5, no. 1, p. 42, 1 2021.

[9] A. L. Guennec, S. Malinowski, and R. Tavenard, "Data Augmentation for Time Series Classification using Convolutional Neural Networks," in *Proc. ECML/PKDD*, 2016.

[10] J. F. Torres *et al.*, "Deep Learning for Time Series Forecasting: A Survey," *Big Data*, vol. 9, no. 1, pp. 3–21, 2021.

[11] K. Benidis *et al.*, "Deep Learning for Time Series Forecasting: Tutorial and Literature Survey," *ACM Comput. Surv.*, vol. 55, no. 6, 121:1–121:36, 2022.

[12] H. Hewamalage, C. Bergmeir, and K. Bandara, "Recurrent Neural Networks for Time Series Forecasting: Current status and future directions," *Int. J. Forecast.*, vol. 37, no. 1, pp. 388–427, 2021.

[13] Z. Ouyang, P. Ravier, and M. Jabloun, "Are Deep Learning Models Practically Good as Promised? A Strategic Comparison of Deep Learning Models for Time Series Forecasting," in *Proc. EUSIPCO*, 2022.

[14] G. Lai *et al.*, "Modeling Long- and Short-Term Temporal Patterns with Deep Neural Networks," in *Proc. ACM SIGIR*, 2018.

[15] D. Salinas *et al.*, "DeepAR: Probabilistic forecasting with autoregressive recurrent networks," *Int. J. Forecast.*, vol. 36, no. 3, pp. 1181–1191, 2020.

[16] S. Bai, J. Z. Kolter, and V. Koltun, "An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling," in *Proc. ICLR*, 2018.

[17] A. Vaswani *et al.*, "Attention is All you Need," in *Proc. NeurIPS*, 2017.

[18] J. Devlin *et al.*, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," in *Proc. NAACL*, 2019.

[19] A. Dosovitskiy *et al.*, "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale," in *Proc. ICLR*, 2021.

[20] T. B. Brown *et al.*, "Language Models are Few-Shot Learners," in *Proc. NeurIPS*, 2020.

[21] H. Zhou *et al.*, "Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting," in *Proc. AAAI*, 2021.

[22] H. Wu *et al.*, "Autoformer: Decomposition Transformers with Auto-Correlation for Long-Term Series Forecasting," in *Proc. NeurIPS*, 2021.

[23] N. Kitaev, Ł. Kaiser, and A. Levskaya, "Reformer: The Efficient Transformer," *Proc. ICLR*, 2020.

[24] S. Li *et al.*, "Enhancing the Locality and Breaking the Memory Bottleneck of Transformer on Time Series Forecasting," in *Proc. NeurIPS*, 2019.

[25] C. Spearman, "The Proof and Measurement of Association between Two Things," *Am. J. Psychol.*, vol. 15, no. 1, pp. 72–101, 1904.

[26] M. Blondel *et al.*, "Fast Differentiable Sorting and Ranking," in *Proc. ICML*, 2020.

[27] R. F. Engle, "Autoregressive Conditional Heteroscedasticity with Estimates of the Variance of United Kingdom Inflation," *Econometrica*, vol. 50, no. 4, pp. 987–1007, 1982.

[28] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," in *Proc. ICLR*, 2014.

[29] A. Paszke *et al.*, "PyTorch: An Imperative Style, High-Performance Deep Learning Library," in *Proc. NeurIPS*, 2019.

[30] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.