Transformer Applying in the Time Series Prediction

Weiheng Li *

Department of Jinan University Birmingham University Joint Institution, Jinan University, Guangzhou, Guangdong, 111000, China

* Corresponding Author Email: liweuheng2023101693@stu2023.jnu.edu.cn

Abstract. Time series prediction is of great significance in various fields such as finance, transportation, and biomedical science. The Transformer architecture and its derivative models, based on their reliance on long-distance relationships and their ability to extract and model complex information, have been widely adopted and deeply studied in time series prediction tasks in recent years, and have achieved considerable results. However, regarding the rapid development and innovation of Transformer models in the field of time series, there has not yet been a systematic classification and comprehensive summary of the research conducted in recent years. This paper reviews the time series prediction methods based on Transformer, and classifies the existing methods into three main paradigms: the widely studied and referenced classic paradigms (such as Informer, PatchTST, Cross former), the innovative Transformer paradigms (integrating signal theory, block theory, and architectural improvements), and the combination paradigms of time series and large language models. The article systematically analyzes the core innovation points, unique mechanisms, and performance levels of various models and directions, and also discusses the limitations of current methods and future research directions. This paper is helpful for researchers and practitioners to gain a systematic reference for understanding and applying time series prediction models based on the Transformer.

Keywords: Time Series, Transformer, Paradigms.

1. Introduction

Time series are widely applied in fields such as finance, transportation, and biomedicine. There has been relatively in-depth research on time series prediction. Early ARIMA and its variant, seasonal ARIMA, were proposed in, and concepts such as prediction windows and periodicity were used to construct the models [1]. Additionally, the Prophet model was proposed which decomposes time series into periodic, trend, and special date (holiday) components for modeling [2]. In the field of deep learning, building sequence models using deep learning for continued prediction has become a popular direction, such as the early RNN, which proposed using time backpropagation to update the network [3]. LSTM utilizes the gating mechanism to enable the model to utilize or discard sequence information [4]. GRU simplifies the gating mechanism of LSTM into two gates [5]. Seq2seq proposes a time series prediction model with a decoding encoder architecture e [6].

Because the input of sequence models has numerical order, they have a similar structure to images. Thus, CNN has also been applied to the prediction of time series. LSTNet utilizes convolution for the sequence prediction [7]. The transformer model was proposed by [8], which directly processes the entire sequence with multi-head self-attention, promoting the efficiency of information extraction. This architecture has been extensive usage in the field of time series and this algorithm has given rise to many variants, such as Patchtst [9], Crossformer [10], etc, gradually becoming a very significant sequence model prediction architecture. With the development of large language models based on the Transformer architecture, their powerful performance has begun to be applied to sequence prediction, such as timeLLM [11], etc.

The prediction models in time series have developed and improved rapidly in recent years. However, there has been a lack of classification and summary of the transformer and its various variants and improved models in this field. Therefore, a survey is needed to summarize the models in this field. This paper, based on the recent developments, classifies and summarizes the improvement directions and technical routes of the transformer model and the time series LLM. In the following

content, Section 2 is about the architecture of the transformer. Section 3 explains the classification of the classic models and their core innovations. Section 4 discusses the existing limitations and future prospects. In the end, Section 5 is the conclusion and summary of the entire survey.

2. Introduction to the transformer

2.1. The Key to the Transformer

The original transformer was proposed by [8] and consists of an encoder and a decoder. The core architecture of the decoder is shown in Figure 1. The decoder is similar to the encoder in terms of structures and processing steps, but there are differences in details and there are various variations. This paper takes the structure interpretation of the decoder as an example. The architecture mainly consists of dimension embedding, position embedding, transformer blocks with multi-head self-attention, MLP and layer normalization, and output heads.

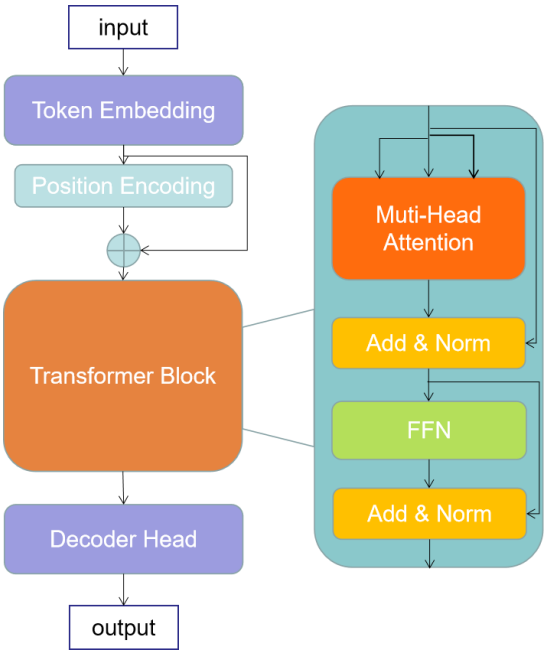

Figure 1. The Structure of Decoder [8]

2.2. Dimension embedding

By a linear layer, it will map the channels of the input data into the internal dimensions of the model. And it is a common operation in deep learning. The Mathematical expression is

$$X_{embeding} = XW \tag{1}$$

Where $X \in R^{L\times C}$ is the input data, $X_{embeding} \in R^{L\times d}$, W is the trainable linear mapping matrix, C is the feature dimension(channels) of each position in the sequence, L is the length of the sequence, and d is the embedding dimension of the model.

2.3. Position encoding

In order to enable the model to distinguish different positions in the sequence, position encoding is added at each position of the sequence. Position encoding can be written as

$$Position_{p,i} = \begin{cases} \sin\left(\frac{p}{10000^{\frac{i}{d}}}\right), i \text{ is even} \\ \cos\left(\frac{p}{10000^{\frac{i-1}{d}}}\right), i \text{ is odd} \end{cases} \tag{2}$$

Where $p = 0,1 \cdots (L-1)$ represents the position, $i = 0,1 \cdots (d-1)$ represents the dimension. Then,

$$X_{input} = X_{embeding} + Position \tag{3}$$

Get the embedding with position information.

2.4. Transformer block

Muti-Head Self-Attention. The multi-head attention mechanism calculates the similarity of features for each part at different positions to obtain the weight of the information in each position, thereby enabling the model to focus on the information at the appropriate positions.

The mathematical formula is expressed as follows

$$Multi_{head(Q,K,V)} = Concat(head1, head2 \cdots headh)W^0 \tag{4}$$

$$headi = softmax\left(\frac{Q_i K_i^T}{\sqrt{d_k}}\right)V_i \tag{5}$$

Where $Q_i = QW_i^Q, K_i = KW_i^K, V_i = VW_i^V$, all feature mapping matrices are $R^{d\times hd}$, $W^0 \in R^{hd\times d}$, these four-mapping matrix all can be trainable.

For self-attention, they are $Q = K = V = X_{input}$, which completely utilizes the information of the sequence to construct a model and focuses on different positions of the sequence. For the transformer model used in sequence auto-regression, its attention mechanism incorporates a mask to ensure that the model only get the preceding information. The mask removes the weights in the upper triangular part of the attention score matrices $softmax(\frac{Q_i K_i^T}{\sqrt{d_k}})$.

Feedforward network. After obtaining the attention information for different sequence positions through self-attention, the data passes through the feed forward network, enabling the model to remember the patterns and knowledge within the data [12]. The feed forward network is typically composed of a two-layer MLP layer, mathematically expressed as

$$ffn(H) = ReLU(HW_1 + b_1)W_2 + b_2 \tag{6}$$

Where $W_1 \in R^{d\times m}, W_2 \in R^{mxd}, b_1 \in R^d, b_2 \in R^m$.

Overall structure of the module. There are layer normalization and residual links at the end of the block. The overall design of the module is given by

$$X_{input} = LayerNorm\left(Multi_{head(X_{input},X_{input},X_{input})} + X_{input}\right) \tag{7}$$

$$X_{input} = LayerNorm\left(ffn\left(X_{input}\right) + X_{input}\right) \tag{8}$$

Here, both the input and output are represented by X, which indicates that after a block, the values are assigned to a new data with identical format, which serves as the input for the next block.

2.5. Decoder Head

Decoder head is always a typical MLP, which is applied to map the data to the characteristic dimensions that meet the application requirements.

3. The Improvement and Application of the Transformer Architecture in Time Series Models

3.1. The classic application paradigm of transformer in time series

To address the issues such as the high computational complexity of attention calculation in time series, the fact that information mainly stems from the relationships between numerical values rather than individual values and the significant differences in features among different channels,

researchers have designed several classic paradigms. Firstly, the relatively lower-level and widely applied model optimization paradigm is introduced.

Informer. The scheme provides a method to simplify the computational complexity. Since self-attention requires the multiplication of all keys and queries (the multiplication of several large matrices), the time complexity and space complexity are $O(L^2)$. It takes a considerable amount of time when calculating long time series. Informer considers that the self-attention matrix is often sparse and has a long-tail distribution. It applies the ProbSparse self-attention mechanism. The core concept of this design is to select the most important queries. The approximate formula is

$$\bar{M}(q_i, K) = \max_j \left\{ \frac{q_i k_j^T}{\sqrt{d}} \right\} - \frac{1}{L_k} \sum_{j=1}^{L_k} \frac{q_i k_j^T}{\sqrt{d}} \tag{9}$$

It reduces the complexity to $O(L \log L)$ and proposes the use of a generative style decoder. Instead of the conventional stepwise auto-regressive method, the input is structured as an initial token (known data) + zero padding (the segment to be predicted). The mechanism of the decoder remains unchanged. During prediction, the entire sequence can be output at once, saving time and reducing the cumulative error caused by auto-regression [13].

PatchTST. The PatchTST model algorithm scheme introduces the concepts of time series segmentation and channel independence. The conventional view of time series is to consider each time point as a sequence element, and all the features (or channels) of that time point constitute a token for the transformer input. However, unlike the application of the transformer in language models where each word has rich and sufficient meanings and is a finite discrete variable and the word is convenient as well as suitable to be directly used as a token, the information of time series stems from the correlations between time points. A single time step is merely an isolated continuous value. Treating point-level values as tokens increases the difficulty for the model to learn the core patterns. Therefore, the model proposes to divide the single-variable time series $X \in R^{1 \times L}$ into small patches. The formula is given by

$$X_{new} \in R^{P \times N}, N = \left\lfloor \frac{L-P}{S} \right\rfloor + 2 \tag{10}$$

The size and step of each sequence in this scheme also optimize the calculation speed of the model by reducing the degree of the sequence. At the same time, the model proposes the channel independence theory, that is, each channel processes independently, which is conducive to the flexible expansion of the model and the adaptation to the different characteristics between each channel. These two ideas have been widely applied in subsequent time series processing [9].

Crossformer. The model proposes a method with relatively high accuracy for processing multi-channel time series. Crossformer retains the design of dividing the series into small patches in PatchTST [9], but designs a two-stage attention (TSA) mechanism for handling information from different channels. For the embedded input $X\_emb \in R^{L \times C \times d_{model}}$, C channels are processed through a classic transformer block with shared parameters separately. Then the output of the time-dimension attention stage and the X_emb are put into the router mechanism. The formula is approximately given by

$$B = MSA_1^{dim}(X_{emb}, Z^{time}, Z^{time}) \tag{11}$$

$$\bar{Z}^{dim} = MSA_2^{dim}(Z^{time}, B, B) \tag{12}$$

Subsequently, through normalization and MLP layers, the features of the channel dimension and the time dimension are effectively extracted. The computational complexity is also reduced. The article proposes a multi-scale transformer structure as well, which constructs information of different scales by merging adjacent patches, effectively capturing the information of different scales of time series [10].

3.2. The innovative transformer paradigm in recent years

This section will be classified and introduce the model based on the improvement directions. The main improvement directions can be categorized as enhancements based on signal theory, enhancements based on patch theory and bold enhancements for normal parts of the transformer.

Combining the signal correlation theory with the transformer architecture many signals possess the characteristics of sequence models. Some signal processing methods, such as the Fourier transform, also have the potential to be applied in time series prediction and have advantages in increasing the model's capacity in capturing and utilizing the information about relationships of different periods in the sequence data.

For instance, combining the Fourier transform and wavelet transform with the attention and decoder-encoder structure of the transformer, FED former designs a module for processing data in the frequency domain. The core idea is to transform the time series into the frequency domain through the discrete Fourier transform (DFT), and randomly select the core key frequency components. The model will process them through the designed frequency enhanced attention (FEA) and convert them back to the time domain through inverse transformation, effectively capturing global information of different periods and handling long-term dependencies [14]. Autoformer combines the Wiener–Khinchin Theorem and the fast Fourier transform (FFT) to replace the self-attention mechanism with an auto-correlation mechanism. Firstly, it applies FFT in the queries and keys to calculate the power spectrum, which is used to filter out the core periods. Then, using the selected periods, it performs Roll operations on the values and calculates the attention weights, which can utilize effectively the periodic information of the sequence [15].

Improvement and optimization of the Patch Theory. The transformer was initially used in text tasks such as translation [8]. When applying the transformer to other fields, the model is required to convert appropriate data groups into tokens, such as in image processing, where ViT [16] provides an effective way to transform an image into a sequence by patching the image. In time series, based on the method of dividing series into patches [9], other improvements and optimizations have been derived, effectively enhancing the capture of information within and between patches. To solve the problem of capturing the information in each patch and the information among different patches, PatchMixer replaces the attention module with a convolution. It performs convolution both on patches within and among each other to extract features [17]. Regarding sequence models based on patches, they are always modeled based on a single patch scale, and multi-scale modeling is incomplete. Patch former proposes different scales of patches to segment the sequence [cite], and then uses a dual attention mechanism with attention within and among patches, and performs weighted gated fusion to obtain the final sequence feature information [18].

A bold questioning and innovation of the classic transformer architecture. Regarding the transformer-based and related structures, there are also some articles that propose novel and less conventional improvement directions or raise doubts about them. For example, iTransformer, for multivariate time series, treating each feature of the sequence as a token it is differs from treating the time dimension of the sequence as the input tokens of the model, which is conducive to solving problems such as representation loss and attention noise under multivariate conditions [19]; Researches based on the permutation invariance of the transformer and other unfavorable properties for sequence prediction raised doubts about the transformer's prediction and pointed out that linear models are significantly superior to models designed based on the transformer on some datasets [20]; CSIDI applied a diffusion model, another commonly used model for generation besides the transformer, to time series prediction, combined with the conditional fractional diffusion model to reverse-complete the sequence information [21].

3.3. Combination of time series and LLM

With the prevalence of LLM in recent years in both academic and commercial fields, applying sequence prediction models for text data to time series has generally become a popular trend. Common approaches include directly utilizing the architecture and parameters of LLM, leveraging

the powerful multi-modal capabilities of LLM or converting continuous time series into discrete tokens similar to the text tokens of LLM.

Directly utilizing LLM. Time series and text data shares certain similarities. Some methods propose to directly modify slightly the structure of LLM or use a pretrained model with a few adjustments for fine-tuning to time series prediction. LLMTIME [11] only adjusts the data preprocessing method to enable the LLM to correctly read the numerical values of the time series and adjusts some non-model internal hyperparameters such as topk and temperature. It directly takes the time series as text and inputs the values to the LLM. The researches shows that the model maintains relatively good results even in the zero-sample scenario [11]. aLLM4TS [22] retaining the overall structure of the LLM while only modifying the embedding layers and output heads to adapt to the time series of patch-based substitution of text tokens. It combines multi-domain samples training and target data fine-tuning to promote the adaptation of LLM to sequential tasks.

Utilizing LLM in combination with text modality data. The features in time series can be derived not only from numerical data but also from the description of the certain time point, such as the date or the event that occurred at that time. The ability of LLM to process text is conducive to the combination of text information and numerical information. [23] proposes to use the date text time stamp as a position embedding in the embedding layer of the LLM, which naturally integrates the sequence information into the time semantics.

By applying the discretization concept, the time series is transformed into discrete tokens. Most models attempt to adapt large models to time series, and the idea of discretizing the time series leans towards treating the time series as discrete text input. [24] proposed to convert the continuous time series into discrete vocabularies through scaling and quantization, and simultaneously construct a vocabulary applicable to time series. It directly reuses the language model framework, enabling the time series to be combined with the large model framework only through simple preprocessing.

4. Current limitations and future prospects

4.1. Limitation

The high computational cost of attention. The time and space complexity of the multi-head self-attention mechanism is $O(L^2)$ which usually requires a lot of memory space and computing time when it is necessary to capture long-distance information.

The Permutation Invariance of the self-attention mechanism in Transformers. Multi-head self-attention, as one of the core components of the Transformer, has invariance in permutation. In simple terms, it is less sensitive to the order and position of the input sequence. This issue is usually alleviated through methods such as position encoding and timestamp embedding, but the effect is limited and cannot completely overcome this defect [20].

The effectiveness of the series of LLM transformer models in time series has been questioned. From [25], LLMs have many advantages in natural language processing, but the discrete token logic of text and the continuous numerical logic of time series have essential differences. It makes it difficult for pretrained LLM on a large amount of text to be efficiently fine-tuned in time series and applied in that context.

4.2. Future Prospects

Strengthening the interpretability and theoretical research of the basic framework. The multi-head attention of the transformer combined with MLP can efficiently capture information, but it lacks interpretability and theoretical support. The black-box processing method is easy to cause insufficient generalization ability, difficulty in identifying the root cause of model problems, which may result in low iteration efficiency and other issues.

By integrating the development of multimodal technologies, we will develop multimodal time series algorithms. Combining time series data with other modal data in this field, such as text, images,

audio and video at each moment and applying the achievements of LLM in the field of multimodality to time series may be an important direction for the innovation of time series.

Design efficient algorithms. To address the high time and space complexity issues of the self-attention mechanism in long sequence inputs, efficient attention algorithms are an important optimization and improvement direction, such as flash attention for auto-regressive models, using the sparse attention mechanism of the attention matrix or simplifying the model structure, etc.

5. Conclusion

In recent years, the transformer has emerged as a powerful and significant model architecture in the field of time series prediction due to its strong feature extraction capabilities and ability to handle the relationships between sequence data. This paper summarizes various models based on the transformer, starting from the classic influential improvement paradigms to the innovative directions of recent years and then to the integration of time series with LLM. It also reviews the cutting-edge technical routes and presents the development trend of this field in recent years. At the same time, it points out the existing limitations, such as the logical differences between the continuous nature of numerical values and the discrete nature of tokens, as well as issues related to computational efficiency, and provides future prospects, such as strengthening theoretical research and exploring multi-modal directions.

References

[1] Bartholomew D J. Time series analysis forecasting and control. 1971.

[2] Taylor S J, Letham B. Forecasting at scale. The American Statistician, 2018, 72 (1): 37 – 45.

[3] Williams R J, Zipser D. A learning algorithm for continually running fully recurrent neural networks. Neural Computation, 1989, 1 (2): 270 – 280.

[4] Hochreiter S, Schmid Huber J. Long short-term memory. Neural Computation, 1997, 9 (8): 1735 – 1780.

[5] Cho K, Van Merriënboer B, Bahdanau D, Bengio Y. On the properties of neural machine translation: encoder-decoder approaches. arXiv preprint, 2014. arXiv: 1409.1259.

[6] Sutskever I, Vinyals O, Le Q V. Sequence to sequence learning with neural networks. Advances in Neural Information Processing Systems, 2014, 27.

[7] Lai G, Chang W C, Yang Y, Liu H. Modeling long-and short-term temporal patterns with deep neural networks. In: The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, 2018: 95 – 104.

[8] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez A N, et al. Attention is all you need. Advances in Neural Information Processing Systems, 2017, 30.

[9] Nie Y, Nguyen N H, Sinthong P, Kalagnanam J. A time series is worth 64 words: long-term forecasting with transformers. arXiv preprint, 2022. arXiv: 2211.14730.

[10] Zhang Y, Yan J. Crossformer: transformer utilizing cross-dimension dependency for multivariate time series forecasting. In: The Eleventh International Conference on Learning Representations, 2023.

[11] Kojima T, Gu S S, Reid M, Matsuo Y, Iwasawa Y. Large language models are zero-shot reasoners. Advances in Neural Information Processing Systems, 2022, 35: 22199 – 22213.

[12] Geva M, Schuster R, Berant J, Levy O. Transformer feed-forward layers are key-value memories. arXiv preprint, 2020. arXiv: 2012.14913.

[13] Zhou H, Zhang S, Peng J, Zhang S, Li J, Xiong H, Zhang W. Informer: beyond efficient transformer for long sequence time-series forecasting. In: Proceedings of the AAAI Conference on Artificial Intelligence, 2021, 35 (12): 11106 – 11115.

[14] Zhou T, Ma Z, Wen Q, Wang X, Sun L, Jin R. Fed former: frequency enhanced decomposed transformer for long-term series forecasting. In: International Conference on Machine Learning, 2022: 27268 – 27286. PMLR.

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

[16] Dosovitskiy A, Beyer L, Kolesnikov A, Weissenborn D, Zhai X, Entertainer T, et al. An image is worth 16x16 words: transformers for image recognition at scale. arXiv preprint, 2020. arXiv: 2010.11929.

[17] Gong Z, Tang Y, Liang J. Patchmixer: a patch-mixing architecture for long-term time series forecasting. arXiv preprint, 2023. arXiv: 2310.00655.

[18] Chen P, Zhang Y, Cheng Y, Shu Y, Wang Y, Wen Q, et al. Pathformer: multi-scale transformers with adaptive pathways for time series forecasting. arXiv preprint, 2024. arXiv: 2402.05956.

[19] Liu Y, Hu T, Zhang H, Wu H, Wang S, Ma L, Long M. itransformer: inverted transformers are effective for time series forecasting. arXiv preprint, 2023. arXiv: 2310.06625.

[20] Zeng A, Chen M, Zhang L, Xu Q. Are transformers effective for time series forecasting? In: Proceedings of the AAAI Conference on Artificial Intelligence, 2023, 37 (9): 11121 – 11128.

[21] Tashiro Y, Song J, Song Y, Ermon S. CSDI: conditional score-based diffusion models for probabilistic time series imputation. Advances in Neural Information Processing Systems, 2021, 34: 24804 – 24816.

[22] Bian Y, Ju X, Li J, Xu Z, Cheng D, Xu Q. Multi-patch prediction: adapting language models for time series representation learning. In: Forty-first International Conference on Machine Learning, 2024.

[23] Liu Y, Qin G, Huang X, Wang J, Long M. AutoTimes: autoregressive time series forecasters via large language models. arXiv preprint, 2024. arXiv: 2402.02370.

[24] Ansari A F, Stella L, Turkmen C, Zhang X, Mercado P, Shen H, Wang Y, et al. Chronos: learning the language of time series. arXiv preprint, 2024. arXiv: 2403.07815.

[25] Tan M, Merrill M, Gupta V, Althoff T, Hartvigsen T. Are language models actually useful for time series forecasting? Advances in Neural Information Processing Systems, 2024, 37: 60162 – 60191.