Proceedings of the 2023 International Conference on Machine Learning and Automation
DOI: 10.54254/2755-2721/38/20230536

Application and challenges of informer model in financial time series prediction: A review

Zequan Zhao

School of Electronic Information, Xijing University, Xian 710123, China

984199027@qq.com

Abstract. Time series prediction has shown excellent performance in various fields in recent years, such as stock prices, weather changes, traffic flow, and other fields. Its application development is becoming increasingly mature, and long series time prediction area of research has gained significant prominence. The excellent performance of deep learning in many models has unleashed the potential and possibility of time series prediction to a certain extent. Based on the above reasons, applying deep learning to the field of time series prediction has become a meaningful research. Therefore, the purpose of this article is to analyze the Informer algorithm model in the area of financial time series prediction and provide a comprehensive literature review on the implementation of Informer models in financial time series. Attempting to investigate and analyze the problems and challenges that Informer models may encounter in the area of financial time series prediction, with the hope of providing innovative inspiration and motivating new forms of knowledge for future workers.

Keywords: Deep Learning, Financial Time Series Prediction, Informer, Prediction Problem Analysis.

1. Introduction

Ever since 2017, the development of Transformer [1] has been aimed at solving the serial computing problem that leads to a sharp decrease in speed. Transformer introduces a multi head autonomous mechanism and adopts parallel computing methods to reduce computation time while reducing performance degradation caused by long-term dependencies. Tranformer has gradually surpassed LSTM as the main method for processing and predicting time series data [2]. Transformer significantly shortens the maximum path, greatly improving the accuracy of neural network prediction of stock closing prices in emerging markets [3]. However, there are some weaknesses, such as local Agnosticism and memory bottlenecks, problems with location information coding, and the disappearance of the top gradient, which leads to the fact that the Transformer model relies too much on high-performance equipment to make efficient use of it under a limited budget. [4] Using Transformer, B&H, RNN, CNN and LSTM, we predict CSI 300 Index, S&P 500, Hang Seng Index and Nikkei 225, and prove the superiority of Transformer. Experiments have shown that Transformer based networks seem to outperform LSTM based networks in stock price prediction. However, the research on Transformer based financial market prediction models is still not in-depth enough.

In 2021, an AGCNT model was proposed to map a primary query to a low-dimensional density diagram structure by generating an adaptive convolution network [5] Lightweight adaptive diagram

DOI: 10.54254/2755-2721/38/20230536

convolutions were designed to extract main query correlations from the generated graph structure, reducing computational complexity. By using the maximum pooling layer to reduce the self-attention output size of the decimation graph of each cascade, the model can maintain the main query relationship in the long sequence, without becoming a memory bottleneck. Improve the inference speed of long-term forecasts by having a stack decoder that generates inference and using the generation interface of the stack decoder to generate all the predictions in one forward operation.

The Informer algorithm model [6] was developed to tackle the issues of high time complexity and memory usage in the vanillaTransformer. It incorporates a ProbSparse self-attention mechanism and extraction operation. The Informer model improves upon the Transformer by focusing on optimizing attention mechanisms and removing the limitations of the encoder-decoder structure. This allows the model to effectively capture correlation patterns between sequences, which was previously a limitation. Experiments using real-world data have shown the Informer's effectiveness in improving the predictive ability of LSTF problems. Compared to the Transformer, the Informer model offers faster attention calculation speed, the ability for the decoder to output all predictions at once, and quicker stacking of encoders. These advantages make the Informer model well-suited for rapidly predicting intraday long series stock prices.

This article provides an overview of Informer based models, with the main contributions as follows:

1.This article discusses the development history of Informer models.

2.In this article, the principle, structure, and attention mechanism of the Informer algorithm model are systematically explained, and a deep analysis of the Informer algorithm is conducted

2. The Development History of Informer

2.1. RNN and LSTM

Initially, in time series problems, researchers utilized Recurrent neural network (RNN) and its specific variant, Long short-term memory network (LSTM). The original LSTM block consisted of unit, input, and output gates, without the inclusion of a forget gate and peephole connection. Certain experiments selectively excluded the output gate, cell bias, or input Activation function. The training process involved a hybrid approach combining real-time recursive learning (RTRL) [7] and backpropagation through time (BPTT) [8]. Only the gradients between cells were able to propagate back in time, while the gradients of other circular connections were truncated. Thus, an exact training gradient was not used in the study. This model has proven to be effective and scalable in solving learning problems related to sequence data [9], achieving relatively accurate short-term predictions. Nevertheless, as the length of time series data increases, the prediction speed of LSTM declines rapidly, rendering it incapable of making effective long series predictions within the given time and limited computing power. Recurrent neural networks suffer from continuous error accumulation, leading to a significant increase in the MSE index [4], as well as problems of gradient disappearance and explosion. Consequently, the traditional Recurrent neural network fails to yield satisfactory results for long sequence problems.

2.2. Transformer

Before studying the Transformer model, researchers used Recurrent neural network (RNN) or Convolutional neural network (CNN) as the basic unit to build a model containing encoder and decoder when doing Machine translation and other things [3]. The effect is relatively good but there is obvious room for improvement. In traditional sequence models, each element of the input sequence is processed one after another, and the state of the previous element is used as input for the subsequent element. This processing method is simple, but at the same time, the model cannot process the input sequence in parallel. As the sequence length increases,the model's training efficiency is decreasing

The Transformer model utilizes a parallel processing approach, employing self attention mechanism for understanding the connections within a sequence. Self attention mechanism is a method used to assess the relationship among elements within a sequence. In the Transformer model, every element is paired with all other elements in the sequence, whereby each element computes a weight to represent its

DOI: 10.54254/2755-2721/38/20230536

correlation with the other elements. By employing the self attention mechanism, the Transformer model can process numerous sequences concurrently, as opposed to following a sequential approach like conventional sequence models. This parallelized processing enhances the efficiency and performance of the Transformer model when dealing with lengthy sequence data.

2.3. Informer

The Informer model is an optimization model proposed based on the three limitations faced by the Transformer model:

(1) The Time complexity and memory usage of each layer are O(L^2), where L is the length of the sequence.

(2) The memory bottleneck in the long input stack layer limits the scalability of the model when receiving long sequence inputs.

(3) The gradual inference in Vanilla Transformer decoding significantly reduces the speed of predicting long output.

SparseTransformer [10], LogSparseTransformer [11], and Longtransformer [12] are considered improved models, but their impact on performance is quite limited. This is particularly true when it comes to predicting stock changes in real-time, especially with regards to the third limitation. As mentioned earlier, as the input sequence gets longer, the MSE score increases, which poses a challenge faced by all LSTM-like models: the longer the sequence, the more likely the results will deviate. Moreover, existing Transformer-based network decoders produce results sequentially, resulting in low efficiency. However, since this study focuses on financial time series, longer prediction times and unsatisfactory results can negatively affect trading. The Informer model effectively addresses these three issues by reducing computational time and achieving accurate predictions for long time series Architecture and principles of Informer algorithm models.

3. Architecture and principles of Informer algorithm models

3.1. Definition of Basic Sequence Problems (LSTF)

By incorporating long-term correlations between spatial and temporal patterns, as well as contextual information and data, LSTF enhances its predictive performance. The latest research has indicated that the performance could be enhanced further by utilizing different variations of Informer's algorithmic model.

In the rolling prediction setting with input $X^t = \{x_1^t, \ldots, x_{L_x}^t | x_i^t \in R^{d_x}\}$, the output is $Y^t = \{y_1^t, \ldots, y_{Ly}^t | y_i^t \in R^{d_x}\}$at time t. LSTF can handle longer output lengths ($d_y \geq 1$).

Many popular models input x_t Encode as hidden state H^t and decode from$H^t = \{h_1^t, \ldots, h_{Lh}^t\}$ to display as γ^t T-inference involves a step-by-step process of "dynamic decoding", where the decoder performs step k based on the previous state h_k^t Calculate the new hidden state and other necessary outputsh_{k+1}^t, and then predict the sequence y in step y_{k+1}^t.

3.2. Architecture of Informer

A original normalized self-attention mechanism is defined using tuple input (i.e., query, key, and value), and the attention weight is calculated by performing the scaling dot product operation A(Q, K,V)=Softmax(QK>T/d)V, where Q,K, and V represent query, key, and value respectively. d represents the reciprocal of the input dimension. To further explore the mechanism of self-attention, assume that qi, ki, and vi represent row i in Q, K, and V, respectively. According to the formula in reference [12,13], the attention weight of the i-th query is defined as a kernel smoothing filter in probabilistic form.

$$A(q_i, K, V) = \sum_j \frac{k(q_i, k_j)}{\sum_l k(q_l, k_l)} v_i = E_{p(k_j|q_i)}[V_j] \quad (1)$$

Where p(k_j|q_j) is the probability derived from k(q_i,k_j)/∑l k(q_i,k_l). The choice of k(q_i,k_j) uses the asymptotic exponent exp(q_i k_j/d). The output obtained through the self-attention mechanism

DOI: 10.54254/2755-2721/38/20230536

is mainly obtained by combining the calculated probabilities p(k_j,q_i). In addition, it is necessary to perform the calculation of the quadratic dot product and the memory use of O (L_Q L_K), which is the main disadvantage of the attention mechanism.

The first term is the logarithm of all the keys qi and Exp, and the second term is the arithmetic average of those keys. If the i th query gets a larger M(qi,K), then it has a higher probability of concern p and is likely to be included in the front-end portion of the long-tail self-concern distribution, where there are advantageous dot product pairs. In addition, ProbeSparse is self-focused by allowing each key to focus on only u major queries.

$$A\ (Q, K,V) = SoftMax(\frac{\bar{Q}K^T}{\sqrt{d}})V \tag{2}$$

Lemma1.

When the qi∈K, condition is met, this equation remains true starting from Lemma 1, and the proposed equation for the maximum average measure is as follows [13]:

$$M(q_i, K) = \max_j \left\{\frac{q_i {k_j}^T}{\sqrt{d}}\right\} - \frac{1}{L_K}\sum_{j=1}^{L_K}\frac{q_i {k_k}^T}{\sqrt{d}} \tag{3}$$

To calculate M(q_i,K), the randomly selected dot product pairs U=LKln LQ are utilized. This implies that the remaining pairs are populated with zeros, and from these, $\overline{Q}$ is chosen as a sparse Top-U. The maximum operator in $M(q_i, K)$ is not affected greatly by zero values and remains numerically stable. In actuality, when performing self-attention computations, the lengths of input queries and keys are typically the same, denoted as L_Q=L_K=L.

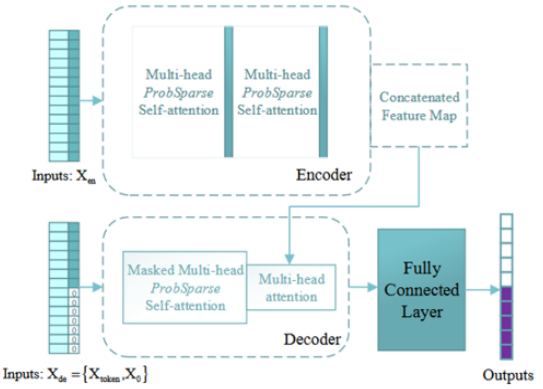

Figure 1. The architecture of the informer algorithm.

3.3. Encoder

Figure 1 shows that the Encoder is given numerous lengthy sequence inputs, while the model replaces the self attention in the Transformer with ProbeSpare self attention. The green trapezoid represents the operation of extracting primary attention through self attention, which effectively reduces the computational complexity and memory usage. Furthermore, the introduction of cascading replicas enhances the robustness.

The purpose of the encoder is to extract strong long-range connections from lengthy sequence inputs. Figure 2 Inspired by the unfolding convolution [14], the "distillation" stage advances from layer j to layer (j+1), as shown below.

$$X_{j+1}^t = MaxPool\left(EIU\left(Conv1d\left(\left[X_j^t\right]_{AB}\right)\right)\right) \tag{4}$$

DOI: 10.54254/2755-2721/38/20230536

An attention block, [·] AB, consists of multi-headed ProbeSparse self-attention and basic operations. The Conv1d (·) function is used with the ELU (·) activation function to perform one-dimensional convolution filtering in the time dimension [15].

Once the researchers finished stacking one layer, they incorporated a maximum pooling layer that covered a width of 2 and reduced the size of Xt slices by half through downward sampling. This resulted in a decrease in overall memory usage to O((2-e) LlogL), where e represents a small value. To strengthen the extraction process's resilience, a duplicate of the primary stack was created using halved input. The output dimensions were aligned, and connections were established by executing one layer at a time.

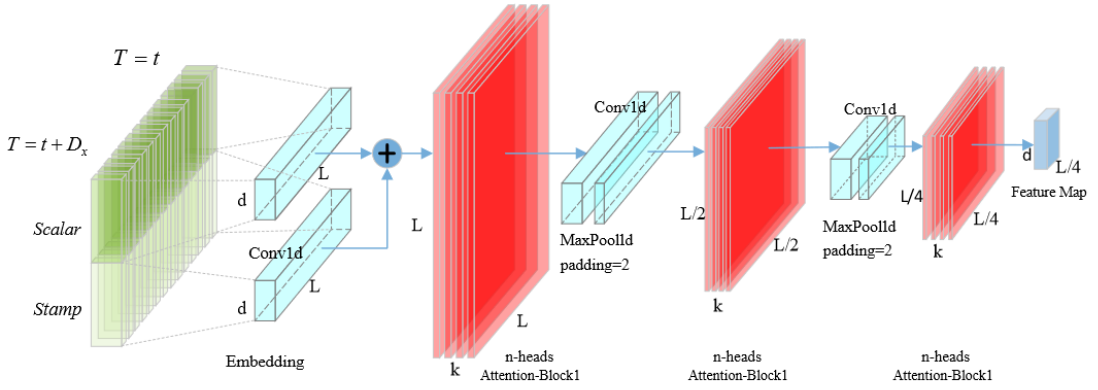

Figure 2. Individual stacks in the encoder.

There is a stack of outputs to obtain the final hidden representation of the encoder to gradually reduce the number of self focusing extraction layers.

3.4. Decoder

The Informer algorithm model utilizes a standard decoder structure comprising two identical multi head attention layers. Yet, in cases of long sequence time prediction, generative reasoning is employed to mitigate decreases in speed. The decoder receives a long sequence of inputs and fills the target element with zeros.With regards to the decoder's implementation principle, the first step is to provide the following vectors to the decoder. In Figure 1, the decoder receives a lengthy sequence of inputs and replaces the target elements with zeros. It then predicts the output elements without any delay.

$$X_{de}^{t} = Concat(X_{token}^{t}, X_{0}^{t}) \in \mathbb{R}^{(L_{token}+L_{y})\times d_{model}} \tag{5}$$

Starting with $X_{token}^{t} \in R^{L_{token}\times d_{model}}$ it serves as the initial marker. $X_{0}^{t} \in R^{L_{y}\times d_{model}}$ is utilized as a substitute for the desired sequence, with the value set at 0. Through the application of the masking dot product set to negative infinity, the multi-head masking attention mechanism is employed during the computation of ProbeSparse self-attention. This technique ensures that each position does not focus on future positions, effectively preventing any autoregressive tendencies. The ultimate output is obtained d_y the fully connected layer, with its dimensionality dependent on whether univariate or multivariate prediction is being performed.

4. Conclusion

The task of predicting stock trends in the field of technical analysis is complicated due to the high level of randomness, dynamism, and interdependence in financial markets. To tackle this challenge, workers utilize the Informer model to predict minute-level fluctuations in stock and market index volatility. This model addresses two main problems. Firstly, there is significant volatility in small-scale trading volume, resulting in rapid changes in stock prices. Secondly, there is an uneven rate of change, causing traditional neural networks to exhibit insensitivity or inconsistent results. By employing the Informer model, we can achieve more accurate predictions of stock trends.

DOI: 10.54254/2755-2721/38/20230536

Furthermore, accurately predicting stock prices is complicated by different trading styles influenced by market conditions, trading rules, and fundamental stock differences. To overcome this issue, it is essential to provide a training set that encompasses comprehensive information and a larger dataset to the network. However, this raises concerns about computational power. Consequently, finding a method to obtain valuable stock information while preserving the original features for preprocessing inventory information and optimizing network structure to reduce computational complexity remain unresolved challenges. Nonetheless, with advancements in artificial intelligence technology and data science, we believe that these issues will soon be effectively addressed.

References

[1] Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need[J]. Advances in neural information processing systems, 2017, 30.
[2] Lin T, Wang Y, Liu X, et al. A survey of transformers[J]. AI Open, 2022.
[3] Malibari N, Katib I, Mehmood R. Smart robotic strategies and advice for stock trading using deep transformer reinforcement learning[J]. Applied Sciences, 2022, 12(24): 12526.
[4] Wang C, Chen Y, Zhang S, et al. Stock market index prediction using deep Transformer model[J]. Expert Systems with Applications, 2022, 208: 118128.
[5] Su H, Wang X, Qin Y. AGCNT: Adaptive Graph Convolutional Network for Transformer-based Long Sequence Time-Series Forecasting[C]//Proceedings of the 30th ACM International Conference on Information & Knowledge Management. 2021: 3439-3442.[23] Asteriou D, Hall S G. ARIMA models and the Box–Jenkins methodology[J]. Applied Econometrics, 2011, 2(2): 265-286.
[6] Zhou H, Zhang S, Peng J, et al. Informer: Beyond efficient transformer for long sequence time-series forecasting[C]//Proceedings of the AAAI conference on artificial intelligence. 2021, 35(12): 11106-11115.
[7] Robinson A J, Fallside F. The utility driven dynamic error propagation network[M]. Cambridge: University of Cambridge Department of Engineering, 1987.
[8] Werbos P J. Generalization of backpropagation with application to a recurrent gas market model[J]. Neural networks, 1988, 1(4): 339-356.
[9] Greff K, Srivastava R K, Koutník J, et al. LSTM: A search space odyssey[J]. IEEE transactions on neural networks and learning systems, 2016, 28(10): 2222-2232.
[10] Child R, Gray S, Radford A, et al. Generating long sequences with sparse transformers[J]. arXiv preprint arXiv:1904.10509, 2019.
[11] Li S, Jin X, Xuan Y, et al. Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting[J]. Advances in neural information processing systems, 2019, 32.
[12] I. Beltagy, M. E. Peters, and A. Cohan, "Longformer: The long-document transformer," arXiv preprint arXiv:2004.05150, 2020.
[13] Clevert, D.-A.; Unterthiner, T.; Hochreiter, S. Fast and accurate deep network learning by exponential linear units (elus). arXiv
[14] Tsai Y H H, Bai S, Yamada M, et al. Transformer dissection: a unified understanding of transformer's attention via the lens of kernel[J]. arXiv preprint arXiv:1908.11775, 2019.
[15] Devlin J, Chang M W, Lee K, et al. Bert: Pre-training of deep bidirectional transformers for language understanding[J]. arXiv preprint arXiv:1810.04805, 2018.