Proceedings of the 3rd International Conference on Signal Processing and Machine Learning
DOI: 10.54254/2755-2721/4/20230432

Analysis of different transformer variants

Yisu Li

School of Computing, Wuhan University, Wuhan 430072, China

2020302111413@whu.edu.cn

Abstract. Efficient self-attention models are crucial in applications where long sequences are modeled, such as documents, images, and videos, are often composed of relatively large numbers of pixels or tokens. Therefore, the efficiency of dealing with long sequences is crucial for the broad adoption of Transformers. This paper talks about four transformer variants: Longformer, Transformer-XL, Big Bird, and Star transformer. This paper makes a comparative analysis of the four varieties' theoretical methods, and obtains the advantages of the four methods and their applicable fields. Longformer has lower complexity and can be used for various document-level tasks. Transformer-XL improves the accuracy by addressing context fragmentation. BigBird replaces the Bert-like whole attention mechanism with Block Sparse Attention, which has achieved the SOTA effect on many tasks with long text sequences, such as long text summaries, long text questions, and answers. Star-Transformer reduces the time complexity to O(n) and performs well in the synthetic data set.

Keywords: Transformer, Longformer, Transformer-XL, Big Bird, Star-Transformer.

1. Introduction

Neural networks have achieved good results in sequential modeling, language modeling, machine translation, and other applications, such as RNN [1], LSTM [2], and GRU [3]. "Recurrent" language models and encoder-decoder architectures are making good progress. However, the inherent sequential nature of RNN hinders parallelization among training samples, and for long sequences, memory limitations will hinder batch processing of training samples. So we need a different model to avoid "Recurrent". This is what transformers do. Transformers [4] use a new mechanism which called self-attention and performs better in many fields compared to CNN [5], and RNN.

Transformers are proposed to solve the problem of slow speed caused by serial input and serial codec in RNN, which mainly contains Longformer [6], Transformer-XL [7], BigBird [8], and Star-Transformer[9]. Longformer improves Transformer's traditional self-attention mechanism. Specifically, each token only gives local attention to tokens near a fixed window size. Moreover, Longformer adds a kind of global attention based on the former local attention for specific tasks. There are three new attention mechanisms, Sliding window attention, Dilated sliding window, and Global+sliding window; all three mechanisms can reduce the complexity of traditional self-attention very well.

Before the Transformer-XL, Vanilla Transformer [10] is proposed to deal with the long articles. However, Vanilla Transformer has three main shortcomings. First, the maximum dependent distance between characters is limited by the length of the input, and the model does not see words that

DOI: 10.54254/2755-2721/4/20230432

appeared more than a few sentences ago. Second, there is no context dependency between segments, which makes training inefficient and affects the performance of the model. Third, in the test phase, each time the next word is predicted, the context needs to be rebuilt, and the calculation starts from scratch, which is very slow. The Transformer-XL architecture introduced two innovations to Vanilla Transformer: Recurrence Mechanism and Relative Positional Encoding, which can solve the problems above.

Big Bird adopts a sparse attention mechanism to replace the original full attention mechanism similar to Bert, and reduces the complexity to linearity, which is O(n). The Big Bird can accommodate longer sequences with 16 GB of RAM (8 times the size of the Transformer, or 4096) and get better performance. It has achieved the SOTA effect on many tasks with long text sequences, such as long text summaries, long text questions, and answers. The core idea of Star-Transformer is to simplify the architecture by moving a fully connected topology into a star structure. In a fully connected network, base connections maintain non-local communication and eliminate redundancy. Ring connection embodies the priority of locality, which has the same function as CNN and RNN.

This paper introduces the theoretical ideas of four transformer variants, Longformer, Transformer-XL, BigBird, and Star transformer, and analyses their specific algorithms. Then it talks their respective changes and improvements based on Transformer. Then the paper analyzes the advantages of the four models and the specific problems they can solve. Longformer improves the traditional self-attention mechanism, which achieves the SOTA on two character-level language modeling tasks. Transformer-XL further improves Transformer's ability to model long-term dependencies, and it also solves the context segmentation problem and improves the prediction speed and accuracy of the model. BigBird draws on the sparsification method of graph structure, and reduces the complexity to linearity, which has achieved the SOTA effect on many tasks with long text sequences, such as long text summaries, long text questions, and answers. Star-transformer reduced the parameters significantly with efficiency, and the complexity is also reduced. Star-Transformer is a significant improvement over standard Transformer for medium-size datasets. Finally, this paper concludes that the varieties of transformers perform better when dealing with Long Article tasks.

2. Transformers

Since Transformer was proposed, it has been widely used in computer vision and has achieved the most efficient results on many datasets. However, the original Transformer was not mature enough to handle extremely long sequences, which could easily exceed 512 when dealing with articles. However, if we continuously increase the dimension of the model, the demand for computing resources during training will increase in square order, which is unbearable. Therefore, different Transformer variants have been proposed to improve or circumvent the original defects, such as Longformer, Transformer-XL, BigBird, and Star-transformer.

2.1.Longformer

In Longformer, there are three new attention mechanisms, including sliding window mechanism, cavity sliding window mechanism, and sliding window mechanism integrating global information, corresponding to B, C, and D in figure 1. These three methods all reduce the complexity of traditional self-attention well.

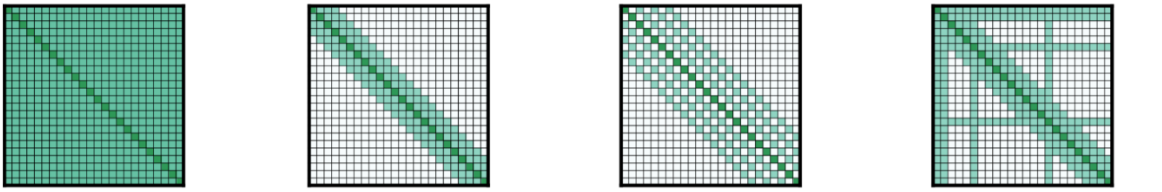

Figure 1. Three new attention mechanisms in Longformer.

DOI: 10.54254/2755-2721/4/20230432

In sliding window attention, for each token, attention is computed for only w tokens in its vicinity, and the complexity is $O\ (n \times w)$, where n is the length of the text. Different window sizes can be applied to different layers of Transformer, depending on the application task.

In the dilated sliding window, when encoding each token, the ordinary sliding window mechanism can only consider the context of length w. The cave-sliding window mechanism (in fact, cave-sliding window is a very early technology in the CV field) can broaden the scope of vision without increasing the computational load. In the sliding window, there will be a gap of size d between two adjacent tokens attended to, so the field of view of each token can reach $d \times w$. The cavity sliding window mechanism performs better than the common sliding window mechanism because of the consideration of more comprehensive context information.

In the Global+sliding window, the language model for the BERT class is implemented slightly differently in Fine-tune. For example, for text classification tasks, the token [CLS] is usually prepended to the entire input. For QA tasks, questions are concatenated with text and typed. In Longformer, a small amount of global attention based on the original local attention can also be added according to the specific task. For example, in the classification task, a global attention will be added at the beginning of [CLS]. In the QA task, global attention is added to all tokens in the question.

The classic self-attention is defined as:

$$Attention(Q, K, T) = softmax(d_k Q K_T) \tag{1}$$

The Attention is calculated after the original input is mapped to the three Spaces of Q, K, V respectively. Global+Sliding Window involves two kinds of Attention. Longformer maps these two types of Attention into two independent Spaces, Namely, Qs, Ks, Vs are used to calculate Sliding Window Attention, and Global Attention was calculated using Qg, Kg, Vg. However, the Attention mechanism proposed above has not been implemented in the current deep learning framework, such as PyTorch/Tensorflow, so three methods of Longformer are implemented and compared, as shown in figure 2.

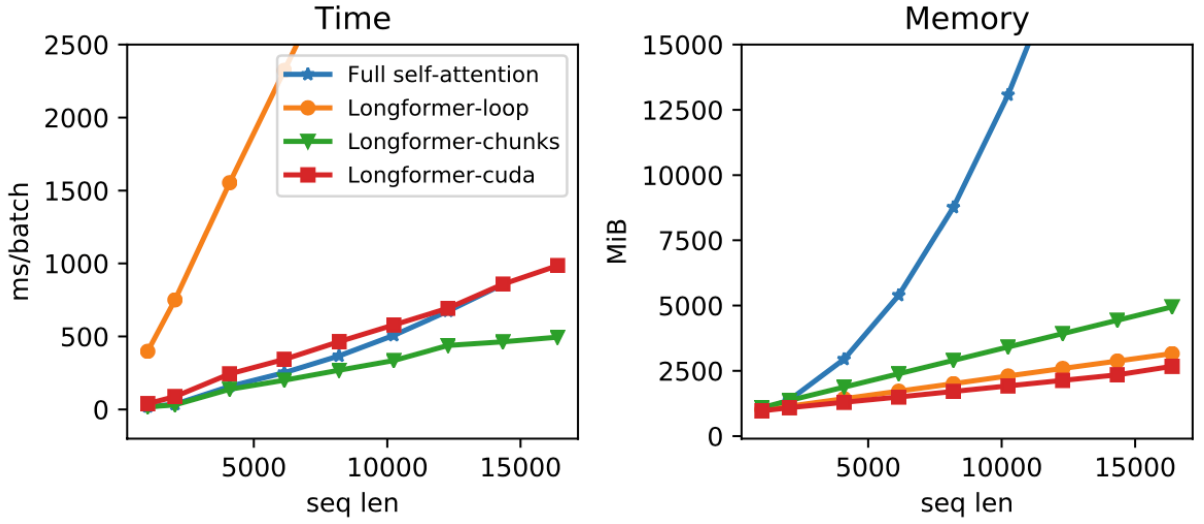

Figure 2. Results of three kinds of Longformer.

Full self-attention is the classic realization of self-attention. Longformer-Loop is a PyTorch implementation that saves memory usage and supports Dilated Sliding Window, but this implementation is too slow, so it can only be used in the test phase. Longformer-Chunks does not support Dilated Sliding Window and is used in the pre-training/fine-tuning phase. Longformer-CUDA is a CUDA kernel method implemented by the author using TVM. As you can see, Longformer can achieve better spatiotemporal complexity.

DOI: 10.54254/2755-2721/4/20230432

2.2.Transformer-XL

Vanilla Transformer is the algorithm that goes between Transformer and Transformer-XL, so before Transformer-XL, we will talk about Vanilla Transformer. The model uses the Decoder structure in Transformer, so figure 3 is one-way.

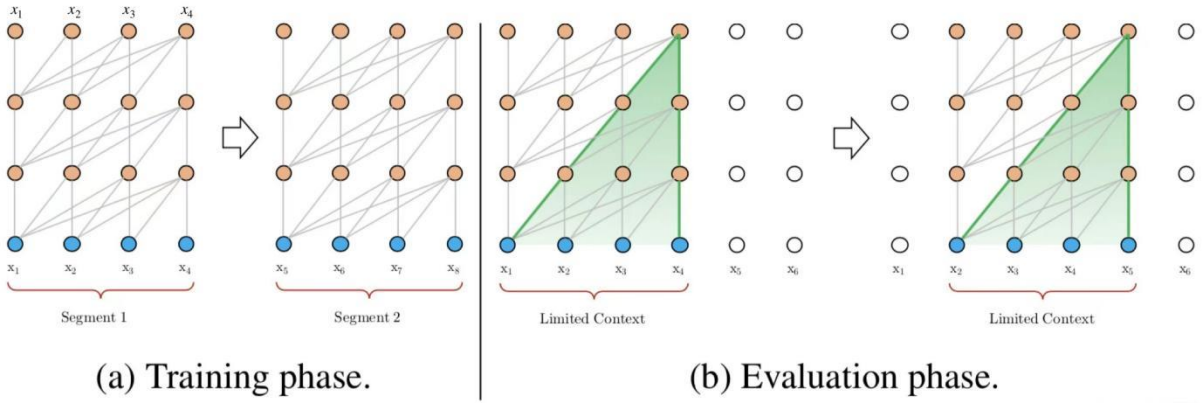

Figure 3. Model of Vanilla Transformer.

Vanilla Transformer has three shortcomings. The first one is length limited. The maximum dependent distance between characters is limited by the length of the input, and the model does not see words that appeared more than a few sentences ago. Second, the semantics between upper and lower segments are not considered. There is no context dependency between segments, which makes training inefficient and affects the performance of the model. The last one is slow reasoning speed. In the test phase, each time you predict the next word, the context is rebuilt, and the calculation starts from scratch, which is very slow.

To improve the performance of the Vanilla Transformer, Transformer-XL architecture introduces two innovations to Vanilla Transformer: Recurrence Mechanism and Relative Positional Encoding. Compared to Vanilla Transformer, another advantage of the Transformer-XL is that it can be used for word-level and character-level language modeling.

Transformer-XL is still modeled piecewise, but it is essentially different from Vanilla Transformer in that it introduces a loop mechanism between segments, so that the current segment can use the information of previous segments to achieve long-term dependencies when modeling. It's shown in figure 4.

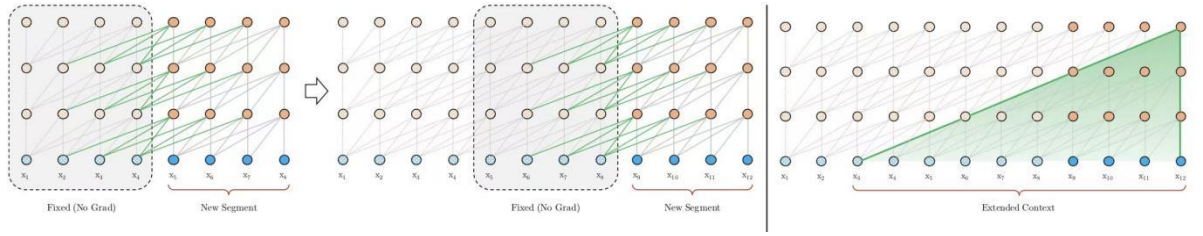

Figure 4. Model of Transformer-XL.

During the training phase, each hidden layer receives two input types as the subsequent segments are processed. First, the output of the preceding node of the segment, which is the same as Vanilla Transformer (grey line above). Second, the output of the nodes in the previous segment (the green line in the figure above) enables the model to create long-term dependencies. This part of output is conducted by cache mechanism, so it will not be involved in the gradient calculation.

In the forecasting phase, when predicting $x11$, we take $[x1, x2, x3 \ldots x9, x10]$, which are predicted before and you predict it. Similarly, when forecasting $x12$, it is directly calculated based on

DOI: 10.54254/2755-2721/4/20230432

$[x1, x2, x3 \ldots x10, x11]$ instead of having to recalculate a fixed number of previous words every time forecasting a word like Vanilla Transformer.

Based on the classical relative position coding, a new relative position coding is proposed. In standard Transformer, the formula of the attention score of query vector $q_i$and key vector k_j is shown in (2).

$$A_{i,j}^{abs} = q_i^T k_j = \underbrace{E_{x_i}^T W_q^T W_k E_{x_j}}_{(a)} + \underbrace{E_{x_i}^T W_q^T W_k U_j}_{(b)} + \underbrace{U_i^T W_q^T W_k E_{x_j}}_{(c)} + \underbrace{U_i^T W_q^T W_k U_j}_{(d)} \quad (2)$$

Then make some changes to the above formula, and get (3).

$$A_{i,j}^{rel} = \underbrace{E_{x_i}^T W_q^T W_{k,E} E_{x_j}}_{(a)} + \underbrace{E_{x_i}^T W_q^T W_{k,R} R_{i-j}}_{(b)} + \underbrace{u^T W_{k,E} E_{x_j}}_{(c)} + \underbrace{v^T W_{k,R} R_{i-j}}_{(d)} \quad (3)$$

First, the absolute position encoding p_j in (b) and (d) is replaced by the relative position encoding R_{i-j}, where R adopts the sinusoid encoding matrix that does not require training in standard Transformer. Second, two learnable parameters, u and v, are introduced to replace U_i, and W_q in (c) and (d), respectively. Because the query vector is the same for all the query locations; That is, the attention bias for different words should be consistent regardless of the query location. Finally, W_k is split into two weight matrices $W_{k,E}$, and $W_{k,R}$, namely the content-based key vector and the position-based key vector respectively.

After modification, each part of the new formula has its meaning. (a) represents content-based addressing (without considering location coding); (b) represents positional deviation concerning content; (c) represents global content bias (measuring critical importance at the content level); (d) represents the global position bias (measuring the significance of the key at the relative position level).

Transformer-XL has emerged to further improve Transformer's ability to model long-term dependencies. Its core algorithm contained two parts: segment-level recurrence mechanism and Relative positional encoding mechanism. The improvements Transformer XL offers include three parts. The first one is the ability to capture long-term dependencies; The second one is that the context segmentation problem is solved, And; the third one is the improvement in the prediction speed and accuracy of the model. State of the art language modeling results are implemented on several different datasets (large/small, character level/word level, etc.). Combining two key concepts of deep learning -- loop and attention -- allows the model to learn long-term dependencies and may be extended to other deep learning areas where this capability is needed, such as audio analysis (e.g., speech data with 16K samples per second).

2.3.BigBird

BigBird's attention mechanic is a full attention mechanic, which is similar to BERT's, so it's not better than BERT's attention mechanic; it's more efficient. In BERT's attention mechanism, the storage is quadratic in the sequence length. In the case of long text, the storage requirement is already unbearable. BigBird Block Sparse Attention is intended to solve this problem. In other words, we should replace BERT's full attention mechanism with Block sparse attention when calculating ∞ times on ∞ length sequences.

Figure 5 is a schematic diagram of Big Bird's sparse Attention, which mainly includes three parts of Attention, Random Attention, Window Attention, and Global Attention. These three attention parts are combined to obtain the Big Bird attention matrix A, as shown in (d). If $A(i,j) = 1$, token i needs to calculate the attention value with token j.

DOI: 10.54254/2755-2721/4/20230432

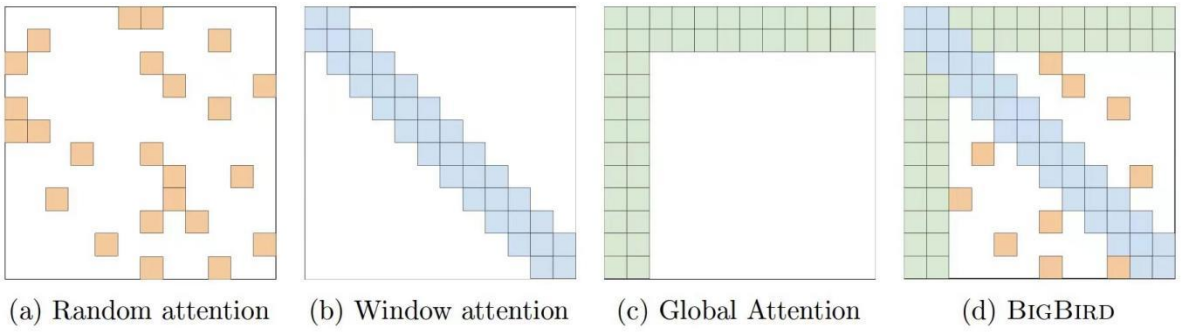

Figure 5. BigBird's sparse attention.

In Random Attention, as shown in (a) above, for each token i, r tokens are randomly selected to calculate Attention. In Window Attention, the sliding Window represents the range of Attention calculation, as shown in (b) above. It mainly captures the local information of token i, and the Window size is w. In Global Attention, as shown in (c), BigBird sets g tokens as global tokens, and all tokens calculate the Attention Score with the global token. There are two ways to define a global token. The first is BigBird-ITc, which selects G tokens as the global token in the original sequence. The second is Bigbird-etc, which adds g extra tokens, similar to [CLS] tokens in BERT.

On today's hardware, such as GPU, a model like BERT combines query tokens with all key tokens to compute the full attention mechanism. Attention computes the square matrix multiplication of sequence length, so it consumes computing resources. However, if sliding attention, global attention, and random attention are combined, and the calculation method is reduced to sparse matrix multiplication, the calculation is easier. BigBird combines Random Attention, Sliding Attention and Global Attention with the block sparse Attention mechanism.

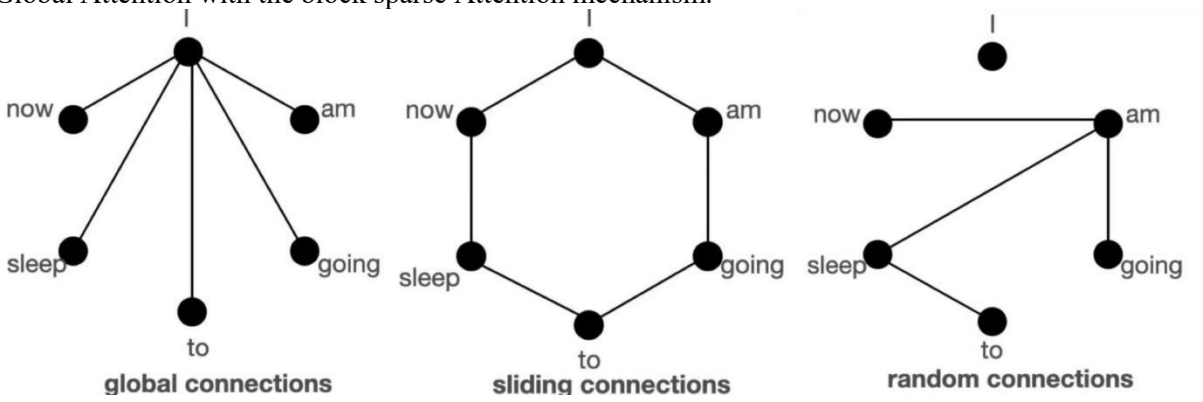

Figure 6. Three connections in BigBird.

Figure 6 shows global, sliding and random connections. Each node represents a word and each line represents attention. If there is no connection between two words, it means there is no attention calculation between the two words. BigBird Block Sparse Attention combines the ten connections of sliding, global and random. The normal attention mechanism is all 15 connections with six nodes. Normal attention is equivalent to all words being global attention.

In normal attention, the model can interact with each word's information directly with any other word in one layer. For example, if the model needs two words, "going" and "now", to be related to each other, they need to be directly connected in one layer.

In Block Sparse Attention, when nodes (or words) of the model need to share information with, the information has to pass through other nodes in the shortest path. For example, when "going" needs to be connected to "now", if sliding attention is considered, the information between "going" and "now"

DOI: 10.54254/2755-2721/4/20230432

must be received by "now" by going -> am -> I -> now. Therefore, Sparse Attention needs several layers of "Block Sparse" to capture all the combined information in the sequence effectively, while Normal Attention can capture all the combined information in only one layer. In extreme cases, such as when two words are furthest apart, the number of layers required is as many as the number of words entered. When joining a Global Connection, information can be passed through going -> I -> now. If you join Random Connections, information can be passed through going -> am -> now. So with the participation of Global Connections and Random Connections, information can be transmitted between tokens more effectively. If there are already many short enough connections when we have a lot of global tokens, then Random connections are no longer needed. By setting num Random Tokens = 0, you get a BigBird model of this variant, also known as ETC.

Since BigBird can now handle sequences up to eight times as long, it can be used for natural language processing tasks such as summaries and question answering systems for longer document formats. In creating BigBird, the researchers also tested its performance on these tasks and witnessed "state of the art results." Since BigBird outperforms BERT in natural language processing, it also makes sense to use this newly established, more efficient model to optimize Google's search result queries.

2.4.Star-Transformer

In Vanilla Transformer, each Token can interact directly with any Token. Tokens are composed of topologies in a fully connected manner, a bit like early computer networks. This kind of interaction is not reasonable enough, and it would be better to change Vanilla Transformer's fully connected topology to a star structure.

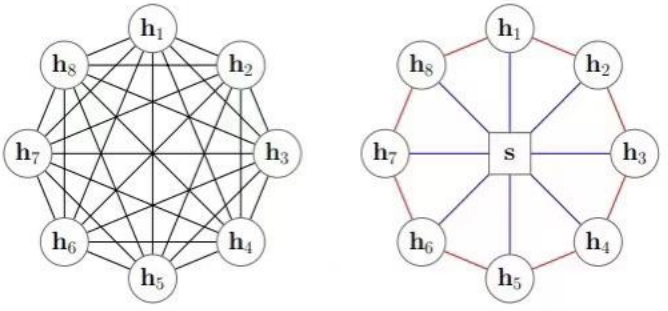

Figure 7. Models of Star-Transformer.

As shown in figure 7, there are two types of nodes in the star-Transformer's Star structure. One is Satellite Nodes, which is similar to terminals in a computer network. Each of the Satellite nodes is connected to relay nodes. In the sequence representation, a Token represents a Satellite node. Another one is Relay Node, similar to a repeater in a computing network, it acts as a bridge for Token interaction.

At the same time, there are two types of links in the structure. The first one is Radical Connections, which links between terminals and relays. With relay nodes, each Token can be linked to any non-neighbour node in a two-step process. The second one is Ring Connections. The left and right adjacent tokens are linked, and the first Token needs to be linked to the last Token to form a Ring. The role of a ring link is similar to that of a CNN or a two-way LSTM.

Compared with Vanilla Transformer, there are two main differences. The first one is that the star model replaces full connection, non-neighbour tokens must pass through the relay node, and two steps update to interact with each other. The number of interactions is reduced, and for some important features, the representation of relay nodes is more concentrated. The second is that neighbour nodes interact directly, and non-neighbour interaction costs a lot. Therefore, Star-Transformer can learn local information more easily than Vanilla Transformers.

DOI: 10.54254/2755-2721/4/20230432

Star-transformer proposes a cycle update method based on the Time step: each Token(H) is initialized by embedding, and the relay node is initially the average value of each Token. Each Token is in turn updated by the Attention mechanism. It is worth noting that K and V in Attention are connected by several vectors. That is, the current value of H is jointly determined by the embedding of the previous node in the last round, the hidden state of this node in the last round, the hidden state of the next node in the last round, the embedding of this node, and the relay node in the last round. Each update is directly connected with the embedding, equivalent to opening a high way with the embedding. After each satellite node has updated its ratio, the relay node is updated. After multiple rounds of updates, Start-Transformer uses $sT + Max(HT)$ as the feature representation of the sequence. At the same time, each H can represent a token independently, which is directly used in the sequence annotation task. The renewal process is shown in figure 8.

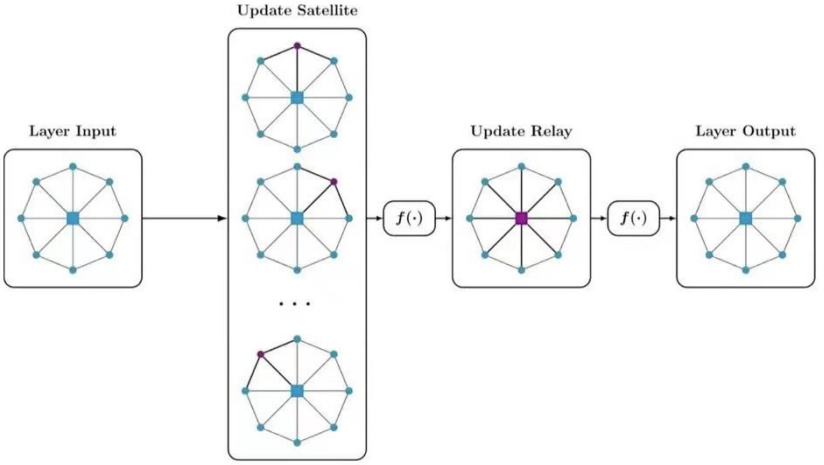

Figure 8. The renewal process of Star-Transformer.

The core idea of Star-Transformer is to simplify the architecture by moving a fully connected topology into a star structure. In a fully connected network, non-local communication is maintained, and redundancy is eliminated. Ring connection embodies the priority of locality and has the same function as CNNs/RNNs. The implementation of the star transformation is completely based on the same attention mechanism as the standard Transformer, which is simpler and more suitable for parallel computing. Due to their better parallelism and lower complexity, Star converters are faster than RNNs or transformers, especially for modeling long sequences.

3. Discussion

Each Transformer variant has proposed different improvement methods for the purpose of improving Transformer efficiency. Longformer improves Transformer's traditional self-attention mechanism. Specifically, each token only gives local attention to tokens near a fixed window size. Moreover, Longformer adds a kind of global attention based on the former local attention for specific tasks. Transformer-XL proposes a segment-level recurrence mechanism and a Relative positional encoding mechanism. BigBird uses Random Attention, Window Attention, and Global Attention to get a Sparse attention mechanism. Star-transformer simplifies the architecture by moving a fully connected topology into a star structure. Through these changes, four transformer variants all improves efficiency. Longformer achieves SOTA on two character-level language modeling tasks. In addition, when using Longformer's attention method to continue pretraining RoBERTa, the performance of the trained language model is better than RoBERTa in multiple long document tasks. Transformer-XL improves the ability to capture long-term dependence, solves the context segmentation problem, and improves the prediction speed and accuracy of the model. BigBird can accommodate longer sequences with 16 GB of RAM (8 times the size of the Transformer, or 4096) and get better performance. It has achieved the SOTA effect on many tasks with long text sequences, such as long text summaries, long text

DOI: 10.54254/2755-2721/4/20230432

questions, and answers. Star-transformer can reduce the parameters significantly with efficiency, and also reduce the complexity. At the same time, it performs better than the traditional transformer on medium-size datasets.

4. Conclusion

This paper focuses on the theoretical model analysis of the four Transformer variants and expounds on its basic algorithm ideas. The four variants solve or alleviate the shortcomings of the basic Transformer model in different aspects. Longformer decreases the complexity and can be used for various document-level tasks. Transformer-XL improves the accuracy and prediction speed. BigBird achieves the SOTA effect on many tasks with long text sequences. Star-Transformer performs better on medium-size datasets.

References

[1] Hou B J and Z H Zhou 2018 Learning with Interpretable Structure from RNN
[2] Sutskever I O Vinyals and Q V Le 2014 Sequence to Sequence Learning with Neural
[3] Networks NIPS MIT Press
[4] Mikolov T et al 2015 Recurrent neural network based language model Interspeech Conference of the International Speech Communication Association Makuhari Chiba Japan September DBLP
[5] Vaswani A et al 2017 Attention Is All You Need arXiv arXiv
[6] Kim Y 2014 Convolutional Neural Networks for Sentence Classification Eprint Arxiv
[7] Zhang P et al 2021 Multi-Scale Vision Longformer: A New Vision Transformer for High-Resolution Image Encoding
[8] Dai Z et al 2019 Transformer-XL: Attentive Language Models beyond a Fixed-Length Context Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics
[9] Zaheer M et al 2020 Big Bird: Transformers for Longer Sequences
[10] Guo Q et al 2019 Star-Transformer Proceedings of the 2019 Conference of the North 2019
[11] Al-Rfou R et al 2018 Character-Level Language Modeling with Deeper Self-Attention.