Adaptive Attention for Sparse-based Long-sequence Transformer

Xuanyu Zhang, Zhepeng Lv and **Qing Yang**
Du Xiaoman Financial
{zhangxuanyu,lvzhepeng,yangqing}@duxiaoman.com

Abstract

Recently, Transformers have been widely used in various fields and have achieved remarkable results. But it is still difficult for Transformer-based models to process longer sequences because self-attention in them scales quadratically with the sequence length. Although some models attempt to use sparse attention to reduce computational complexity, hand-crafted attention patterns are unable to select useful tokens adaptively according to the context. Thus, in this paper, we propose a novel efficient Transformer model with adaptive attention, A^2-Former, for long sequence modeling. It can select useful tokens automatically in sparse attention by learnable position vectors, which consist of meta position and offset position vectors. Because the learnable offset position is not an integer vector, we utilize the interpolation technique to gather corresponding vectors from the input embedding matrix by discrete indexes. Experiments on Long Range Arena (LRA), a systematic and unified benchmark with different tasks, show that our model has achieved further improvement in performance compared with other sparse-based Transformers.

1 Introduction

Transformer-based models (Vaswani et al., 2017) have achieved state-of-the-art performance on a wide variety of natural language processing tasks (Devlin et al., 2019; Liu et al., 2019; Yang et al., 2019). It is also gradually applied to other research fields such as speech and computer vision (Dong et al., 2018; Li et al., 2019; Zhang et al., 2020; Dosovitskiy et al., 2021; Zhu et al., 2021; Touvron et al., 2021). Although self-attention module, the core component in Transformer, can capture global contexts from the whole sequence, the time and memory complexity are both quadratic to the sequence length. Especially when facing longer sequences, Transformer becomes more difficult to process them efficiently and effectively.

Recently, a wide spectrum of efficient Transformers (Child et al., 2019; Ho et al., 2019; Rae et al., 2020; Zhao et al., 2019; Kitaev et al., 2020; Tay et al., 2020; Beltagy et al., 2020; Choromanski et al., 2020; Wang et al., 2020; Zaheer et al., 2020; Roy et al., 2021; Xiong et al., 2021; Tay et al., 2021a; Ma et al., 2021; Chen, 2021; Zhu and Soricut, 2021; Liu et al., 2022) have been proposed to tackle the problem of efficiency, which can be roughly divided into the following directions: sparse attention, low-rank and kernel methods. Because sparse-based attention is intuitive and interpretable in addition to efficiency, we focus on this method in this paper. It usually utilizes some strategies or patterns to limit the number of tokens involved in the attention calculation. Different from traditional sparse Transformer (Martins and Astudillo, 2016; Correia et al., 2019; Peters et al., 2019) with different softmax and pattern-related quadratic computation, recent works mainly adopt sliding windows to achieve linear complexity. For example, Longformer (Beltagy et al., 2020) employs an attention pattern that combines local windowed attention with task-motivated global attention while also scaling linearly with the sequence length. BigBird (Zaheer et al., 2020) incorporates random attention (queries attend to random keys) besides global tokens and local sliding windows. However, these hand-crafted attention patterns mentioned above are usually selected empirically or randomly. It is not an ideal solution for modeling long sequences. How to adaptively select useful tokens for sparse attention according to the context is still an important problem to be considered.

To address these issues, we propose A^2-Former with adaptive attention to model longer sequences in this paper. It can select useful tokens automatically in sparse attention by learnable position vectors, which consist of meta position and offset position vectors. Because each element in the learnable offset position vector is not an integer, we utilize linear interpolation to gather discrete vectors from original the input embedding matrix. Position visualization further shows that traditional attention patterns are not enough to cover the valuable positions automatically selected by models. Experiments on Long Range Arena, a systematic and unified benchmark with different tasks, show that our model has achieved further improvement in performance compared with other sparse-based Transformers.

Overall, the main contributions are as follows:

- We propose a novel efficient Transformer, A^2-Former, which replaces hand-crafted attention patterns with learnable adaptive attention in sparse attention. Besides, position visualization (Figure 3) further shows that traditional attention patterns are not enough to cover the useful positions automatically selected by models.

- We adopt an interpolation technique to help the model gather discrete positions with a continuous weight matrix. By combining the meta position and generated offset position, the position of tokens can be selected dynamically according to the context.

- Experiments on different long sequence tasks validate the effectiveness of our model. Especially, compared with the previous best sparse attention model, BigBird (Zaheer et al., 2020), our model achieves better results.

2 Related Work

Recently, Transformer (Vaswani et al., 2017) and its variants (Devlin et al., 2019; Radford et al., 2018; Liu et al., 2019; Yang et al., 2019) have been widely used in natural language processing (OpenAI, 2023; Zhang et al., 2023; OpenAI, 2022; Zhang and Yang, 2021a; Zhang, 2020; Zhang and Wang, 2020; Zhang, 2019), computer vision (Dosovitskiy et al., 2021; Zhu et al., 2021; Touvron et al., 2021; Zhang and Yang, 2021b), speech (Dong et al., 2018; Li et al., 2019; Zhang et al., 2020) and other domains. To improve computational and memory efficiency, a dizzying number of efficient Transformers (Child et al., 2019; Ho et al., 2019; Rae et al., 2020; Zhao et al., 2019; Kitaev et al., 2020; Tay et al., 2020; Beltagy et al., 2020; Choromanski et al., 2020; Wang et al., 2020; Zaheer et al., 2020; Roy et al., 2021; Xiong et al., 2021; Tay et al., 2021a; Ma et al., 2021; Chen, 2021; Zhu and Soricut, 2021; Liu et al., 2022) have been proposed recently, which can be roughly divided into two directions: sparse attention, low-rank and kernel methods.

Sparse attention methods usually limit the field of view to fixed or random patterns. These patterns can also be used in combination. For example, Sparse Transformer (Child et al., 2019) combines stride and fixed factorized attention by assigning half of its heads to the pattern for reducing the complexity of a traditional Transformer. Longformer (Beltagy et al., 2020) integrates a windowed local-context self-attention and task-oriented global attention that encodes inductive bias about the corresponding task. BigBird (Zaheer et al., 2020) incorporates random attention besides global attention and local window attention. Random attention means that each query attends to a small number of random keys. However, it is still difficult for these hand-crafted, random or combined attention patterns to select valuable pairs in the sparse attention calculation. Different from them, our proposed sparse attention mechanism can automatically and efficiently learn the position that should be selected and calculated. Especially, our model is also different from traditional sparse Transformer (Martins and Astudillo, 2016; Correia et al., 2019; Peters et al., 2019). They only focus on sparse softmax and its threshold and still require quadratic computation to determine the sparsity pattern.

Low-rank and kernel methods are the other solutions to improve the efficiency of Transformer. Low-rank methods usually assume a low-rank structure in the self-attention matrix. For example, Linformer (Wang et al., 2020) decomposes the original scaled dot-product attention into multiple smaller attentions through linear projections, such that the combination of these operations forms a low-rank factorization of the original attention. And kernel methods rewrite the self-attention mechanism through kernelization. For example, Performer (Choromanski et al., 2020) scales linearly rather than quadratically in the number of tokens in the sequence, which is characterized by subquadratic space complexity and does not incorporate any sparsity pattern priors. Different from these mathematical and theoretical methods, our proposed method is still based on sparse attention but focuses more on how to find and learn attention patterns effectively and efficiently.

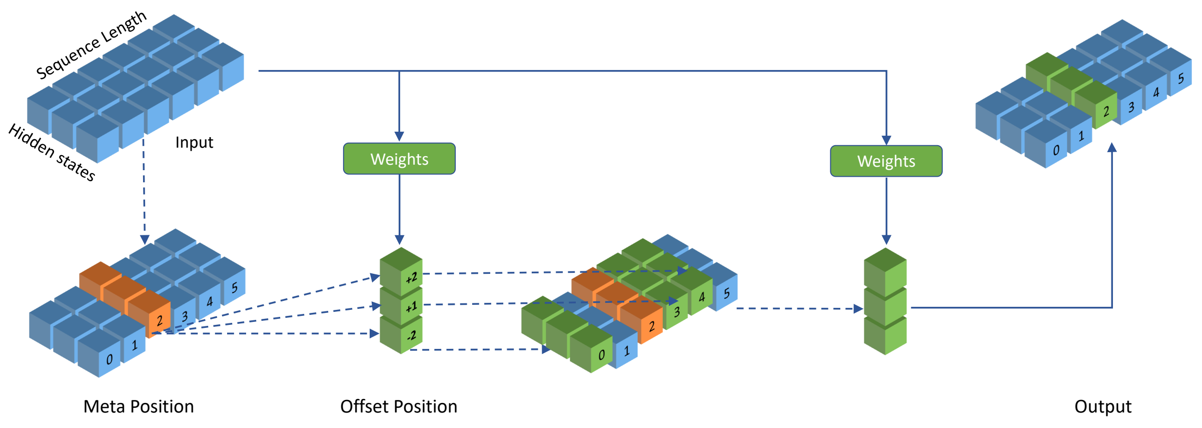

Figure 1: The structure of adaptive attention.

3 Methodology

3.1 Preliminary

Suppose the input is $\mathbf{x} \in \mathbb{R}^{L \times H}$. $q \in \Psi_q, k \in \Psi_k, v \in \Psi_v$ index query, key and value element in Transformer, respectively. L is the sequence length and H is the dimension of hidden states. Thus self-attention in vanilla Transformer can be calculated by

$$\text{Attn}(\mathbf{x}_q, \mathbf{x}) = \sum_{\substack{k \in \Psi_k \\ v \in \Psi_v}} \alpha_{qk} \cdot W \mathbf{x}_v, \quad (1)$$

where W is the learnable weight for $\mathbf{x}_v$. The attention weights $\alpha_{qk} \propto \exp\{\frac{x_q^T W'^T \, W'' x_k}{\sqrt{H}}\}$, where $\mathbf{W}'$ and $\mathbf{W}''$ are learnable weight matrices for $\mathbf{x}_q$ and $\mathbf{x}_k$. The attention weights are normalized as $\sum_{k \in \Psi_k} \alpha_{qk} = 1$, ensuring that they represent the relative importance of each key vector in the set Ψ_k for the query vector $\mathbf{x}_q$.

For sparse attention, we can also express previous models in a unified form. We will only consider the query and key in Transformer in the following discussion. Thus sparse attention can be represented as

$$\text{SparseAttn}(\mathbf{x}_q, \mathbf{x}, \mathbf{p}_q) = \sum_{k=1}^{K} \alpha_{qk} \cdot W \mathbf{x}_{p_{qk}}, \quad (2)$$

where k indexes the sampled keys, and K is the total sampled key number. Because only a small set of keys are utilized in sparse attention, $K \ll L$. $\mathbf{p}_q$ represents the position of K sampled keys for the query $\mathbf{x}_q$. Different models utilize different patterns to select each sampling position $p_{qk} \in \mathbf{p}_q$, such as sliding window or random generation.

Algorithm 1: Adaptive Attention

```
input  : an input matrix $\mathbf{x} \in \mathbb{R}^{L \times H}$;
output : $\text{AAttn}(\mathbf{x}_q, \mathbf{x})$ after adaptive attention;
1  begin
2      Generate the meta position $\hat{p}_q$;
3      for each $q \in [1, L)$ do
4          Calculate the offset position $\beta_{qk}$ via Eq. 4;
5          Calculate the final position $p_{qk}$ via Eq. 5;
6          Rescale each element $p_{qk}$ to $[0, L]$;
7          Round $p_{qk}$ down and up to
             $i = \lfloor p_{qk} \rfloor, j = \lceil p_{qk} \rceil$, respectively;
8          Gather $\mathbf{x}_i, \mathbf{x}_j$ according to the integer vector
             $i, j$ from the input x;
9          Calculate the representation $\mathbf{x}_{\hat{p}_q + \beta_{qk}}$
             according to $\mathbf{x}_i, \mathbf{x}_j$ via Eq. 6;
10         Calculate the attention weights $\alpha_{qk}$ of
             different sampling keys;
11         Calculate the final result $\text{AAttn}(\mathbf{x}_q, \mathbf{x})$ via
             Eq. 3;
12     end
13 end
```

Because our proposed adaptive attention is also based on sparse attention, which can be further refined into the following forms:

$$\text{AAttn}(\mathbf{x}_q, \mathbf{x}) = \sum_{k=1}^{K} \alpha_{qk} \cdot W \mathbf{x}_{\hat{p}_q + \beta_{qk}}, \quad (3)$$

where β_{qk} represents the offset position of the selected key k for the query $\mathbf{x}_q$, $\hat{p}_q$ represents the meta position predefined for each query $\mathbf{x}_q$ according to their absolute index. That is to say, the final position of keys p_{qk} is obtained from the meta position $\hat{p}_q$ and the offset position β_{qk}. Because $\hat{p}_q + \beta_{qk}$ is a float , we adopt linear interpolation to compute $\mathbf{x}_{\hat{p}_q + \beta_{qk}}$. The detailed calculation process will be described in the next subsection.

3.2 Adaptive Attention

As shown in Figure 1, we propose the adaptive attention to learn sampling position dynamically in sparse attention. The pipeline of our proposed adaptive attention is illustrated in Algorithm 1. For convenience, we describe them in the form of iteration rather than batch. We take $L = 6, H = 3, K = 3$ as an example to illustrate the whole process from input to output.

First, we will assign the meta position $\hat{\mathbf{p}}_q = \{\hat{p}_q\}_K$ in Eq. 3 according to the absolute index of the query token. As shown in Figure 1, the meta position is from 0 to 5 for the sentence with 6 tokens. The position of sampling keys will be generated according to the meta position of the query. We will take the orange token (in Figure 1) as a query to obtain the corresponding representation after adaptive attention.

Then, we use a learnable weights $\hat{W} \in \mathbb{R}^{K \times H}$ to obtain the offset position $\beta_q \in \mathbb{R}^K$ for K sampling keys in Eq. 4. As shown in Eq. 5, we can obtain the final position $\mathbf{p}_q \in \mathbb{R}^K$ from original position $\hat{\mathbf{p}}_q \in \mathbb{R}^K$ by combining the meta position and the offset position.

$$\beta_q = \hat{W}\mathbf{x}_q, \tag{4}$$

$$\mathbf{p}_q = \hat{\mathbf{p}}_q + \beta_q, \tag{5}$$

Because the final position $\mathbf{p}_q$ is not an integer vector, it can not be used directly to select sampling keys. Inspired by previous works (Dai et al., 2017; Zhu et al., 2021) in computer vision, we transform bilinear interpolation of two-dimensional images into linear interpolation of one-dimensional text. That is to say, we utilize linear interpolation to gather vectors of corresponding positions. After we rescale each element $p_{qk} \in \mathbf{p}_q$ in $p_{qk} = \hat{p}_q + \beta_{qk}$ to $[0, L]$, we round it down and up to $i = \lfloor p_{qk} \rfloor, j = \lceil p_{qk} \rceil$ respectively ($j - i = 1$). Then we can gather $\mathbf{x}_i, \mathbf{x}_j$ according to the integer i, j from the input x. According to the variation of linear interpolation formula, the final position of sampling keys $\mathbf{x}_{\hat{p}_q + \beta_{qk}}$ can be calculated by

$$\begin{aligned} \mathbf{x}_{\hat{p}_q + \beta_{qk}} &= \frac{p_{qk} - j}{i - j}\mathbf{x}_i + \frac{p_{qk} - i}{j - i}\mathbf{x}_j \\ &= (j - p_{qk})\mathbf{x}_i + (p_{qk} - i)\mathbf{x}_j \end{aligned} \tag{6}$$

Next, we use the learnable matrix α_{qk} to obtain the weights of different sampling keys for the query $\mathbf{x}_q$. Then we can obtain the final weighted representation $\mathrm{AAttn}(\mathbf{x}_q, \mathbf{x})$ in Eq. 3.

Model	ListOps	Text	Retrieval	Image	Pathfinder	Avg
Chance	10.00	50.00	50.00	10.00	50.00	44.00
Transformer	36.37	64.27	57.46	42.44	71.40	54.39
Local Attn.	15.82	52.98	53.39	41.46	66.63	46.06
Linformer	35.70	53.94	52.27	38.56	76.34	51.36
Reformer	37.27	56.10	53.40	38.07	68.50	50.67
Sinkhorn	33.67	61.20	53.83	41.23	67.45	51.39
Synthesizer	36.99	61.68	54.67	41.61	69.45	52.88
Linear Tran.	16.13	65.90	53.09	42.34	75.30	50.55
Performer	18.01	65.40	53.82	42.77	**77.05**	51.41
H-Tran.	**49.53**	78.69	63.99	46.05	68.78	61.41
Sparse Tran.	17.07	63.58	59.59	44.24	71.71	51.24
Longformer	35.63	62.85	56.89	42.22	69.71	53.46
BigBird	36.05	64.02	59.29	40.83	74.87	55.01
Our Model	39.70	**86.14**	**65.94**	**47.57**	71.71	**62.21**

Table 1: Experimental results on five different tasks, i.e., ListOps, Text, Retrieval, Image and Pathfinder. The last four lines are the main sparse attention methods for comparison. (The best model is in boldface and the second best is underlined.)

We can further optimize the complexity for some classification tasks based on sequence level without pre-training. Since sequence level representation is more useful than token level in these tasks, we can convert $\mathbf{x} \in \mathbb{R}^{L \times H}$ to $\mathbf{x}' \in \mathbb{R}^{L' \times H}$ by linear projection, where L' can be set to half of L or even smaller. The detailed performance will be further analyzed in the next section.

4 Experiments

4.1 Datasets

Long-Range Arena (LRA) (Tay et al., 2021b) is a systematic and unified benchmark for the purpose of evaluating sequence models under the long-context scenario, which includes six tasks to assess different capabilities of efficient Transformers like their ability to model relations and hierarchical/spatial structures, generalization capability, etc. These tasks include different domains, such as math, language, image, spatial and so on. Following the original datasets, we use accuracy as the metric for these tasks.

4.2 Implementation Details

Because different tasks have different lengths and characteristics, we use the same hyper-parameters as those described in (Tay et al., 2021b) for a fair comparison. Specifically, the max length is set to 2,000, 4,000, 4,000 for ListOps, Text and Matching task, respectively. The hidden states in attention is set to 512, 256, 128 for ListOps, Text and Matching task, respectively. In our experiments, Adamax (Kingma and Ba, 2014) is used as our optimizer with 0.05 learning rate. The sampling size K for

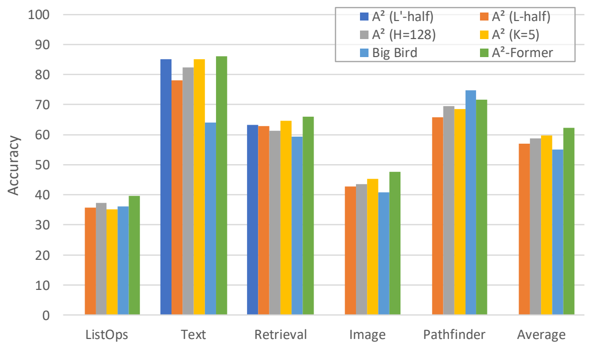

Figure 2: Performance analysis.

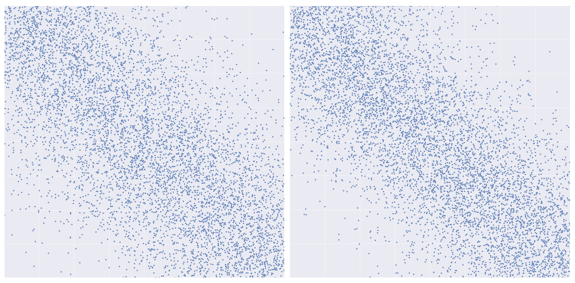

Figure 3: Position visualization.

each token is ten in all the tasks. To prevent overfitting, we use dropout and set it to 0.1. We integrate our attention into the igloo framework (Sourkov, 2018) and run them in Keras with Tensorflow backend on NVIDIA V100 GPU.

4.3 Results

We compare our model with the following state-of-the-art methods as baselines, including sparse attention methods and low-rank and kernel methods. Sparse attention methods include Sparse Transformer (Child et al., 2019), Longformer (Beltagy et al., 2020), Big Bird (Zaheer et al., 2020) and so on. The results on five tasks are summarized in Table 1. It shows that our proposed A^2-Former achieves 62.21 average accuracy, which outperforms the best sparse model based on sliding window, Big Bird (Zaheer et al., 2020), by 7.2%. Thus, the adaptive attention approach proposed in this paper is shown to be superior to traditional hand-crafted, random, or combined patterns in sparse-based Transformer.

4.4 Analysis

As shown in Figure 2, we further analyze the impact of different configurations and parameters on five different tasks. As mentioned above, our proposed A^2-Former has achieved a huge improvement compared to the previous best sparse attention model, BigBird (Zaheer et al., 2020), which proves that even models that combine multiple manual attention patterns is still inferior to the models that learn attention patterns automatically.

We attempt to adjust the maximum input length L to half, change hidden states H to small, and reduce the sampling number K. We can observe that the performance of A^2-Former decreased compared with the original model. specifically, shorter length means less time and content. It is important to find a balance between efficiency and effectiveness according to different tasks. Although the impact of length on some classification tasks based on sequence level is not significant. For adaptive sparse attention, K limits the number of tokens involved in the calculation in each row of the attention matrix, which is also a factor that needs to be balanced.

4.5 Visualization

As shown in Figure 3, we randomly selected two examples for visualization. To study the distribution of positions, we only show the position of the selected tokens in sparse attention matrix. The max length of long sequences is 2000. It is obvious that previous hand-crafted attention patterns, such as sliding window attention, are not enough to cover the positions automatically selected by models. From a general trend, these selected positions are indeed distributed on the diagonal, but to cover these positions, a window size of about half the maximum length is required, which is unacceptable in terms of efficiency.

5 Conclusion

In this paper, we propose a novel sparse-based Transformer, A^2-Former, which replaces hand-crafted attention patterns with learnable adaptive attention in sparse attention. We creatively adopt an interpolation technique to help the model gather discrete positions with continuous position vectors. By combining the meta position and generated offset position, the position of tokens can be selected dynamically according to the context. And position visualization further shows that traditional attention patterns are not enough to cover the useful positions automatically selected by models. Experiments on LRA show that our model has been significantly improved compared with the previous sparse Transformers based on sliding windows.

References

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

Peng Chen. 2021. PermuteFormer: Efficient relative position encoding for long sequences. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10606–10618, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. 2019. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*.

Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, David Belanger, Lucy Colwell, et al. 2020. Masked language modeling for proteins via linearly scalable long-context transformers. *arXiv preprint arXiv:2006.03555*.

Gonçalo M. Correia, Vlad Niculae, and André F. T. Martins. 2019. Adaptively sparse transformers. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2174–2184, Hong Kong, China. Association for Computational Linguistics.

Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. 2017. Deformable convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 764–773.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Linhao Dong, Shuang Xu, and Bo Xu. 2018. Speech-transformer: a no-recurrence sequence-to-sequence model for speech recognition. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5884–5888. IEEE.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2021. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*.

Jonathan Ho, Nal Kalchbrenner, Dirk Weissenborn, and Tim Salimans. 2019. Axial attention in multidimensional transformers. *arXiv preprint arXiv:1912.12180*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. 2020. Reformer: The efficient transformer. *International Conference on Learning Representations*.

Naihan Li, Shujie Liu, Yanqing Liu, Sheng Zhao, and Ming Liu. 2019. Neural speech synthesis with transformer network. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6706–6713.

Yang Liu, Jiaxiang Liu, Li Chen, Yuxiang Lu, Shikun Feng, Zhida Feng, Yu Sun, Hao Tian, Hua Wu, and Haifeng Wang. 2022. Ernie-sparse: Learning hierarchical efficient transformer through regularized self-attention. *arXiv preprint arXiv:2203.12276*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Xuezhe Ma, Xiang Kong, Sinong Wang, Chunting Zhou, Jonathan May, Hao Ma, and Luke Zettlemoyer. 2021. Luna: Linear unified nested attention. *Advances in Neural Information Processing Systems*, 34:2441–2453.

Andre Martins and Ramon Astudillo. 2016. From softmax to sparsemax: A sparse model of attention and multi-label classification. In *International conference on machine learning*, pages 1614–1623. PMLR.

OpenAI. 2022. Chatgpt.

OpenAI. 2023. Gpt-4 technical report.

Ben Peters, Vlad Niculae, and André F. T. Martins. 2019. Sparse sequence-to-sequence models. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1504–1519, Florence, Italy. Association for Computational Linguistics.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. 2018. Improving language understanding by generative pre-training. OpenAI.

Jack W. Rae, Anna Potapenko, Siddhant M. Jayakumar, Chloe Hillier, and Timothy P. Lillicrap. 2020. Compressive transformers for long-range sequence modelling. In *International Conference on Learning Representations*.

Aurko Roy, Mohammad Saffar, Ashish Vaswani, and David Grangier. 2021. Efficient content-based sparse attention with routing transformers. *Transactions of the Association for Computational Linguistics*, 9:53–68.

Vsevolod Sourkov. 2018. Igloo: Slicing the features space to represent sequences. *arXiv preprint arXiv:1807.03402*.

Yi Tay, Dara Bahri, Donald Metzler, Da-Cheng Juan, Zhe Zhao, and Che Zheng. 2021a. Synthesizer: Rethinking self-attention for transformer models. In *International Conference on Machine Learning*, pages 10183–10192. PMLR.

Yi Tay, Dara Bahri, Liu Yang, Donald Metzler, and Da-Cheng Juan. 2020. Sparse sinkhorn attention. In *International Conference on Machine Learning*, pages 9438–9447. PMLR.

Yi Tay, Mostafa Dehghani, Samira Abnar, Yikang Shen, Dara Bahri, Philip Pham, Jinfeng Rao, Liu Yang, Sebastian Ruder, and Donald Metzler. 2021b. Long range arena : A benchmark for efficient transformers. In *International Conference on Learning Representations*.

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Herve Jegou. 2021. Training data-efficient image transformers amp; distillation through attention. In *International Conference on Machine Learning*, volume 139, pages 10347–10357.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Sinong Wang, Belinda Z Li, Madian Khabsa, Han Fang, and Hao Ma. 2020. Linformer: Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768*.

Yunyang Xiong, Zhanpeng Zeng, Rudrasis Chakraborty, Mingxing Tan, Glenn Fung, Yin Li, and Vikas Singh. 2021. Nyströmformer: A nystöm-based algorithm for approximating self-attention. In *Proceedings of the... AAAI Conference on Artificial Intelligence. AAAI Conference on Artificial Intelligence*, volume 35, page 14138. NIH Public Access.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. 2020. Big bird: Transformers for longer sequences. In *NeurIPS*.

Qian Zhang, Han Lu, Hasim Sak, Anshuman Tripathi, Erik McDermott, Stephen Koo, and Shankar Kumar. 2020. Transformer transducer: A streamable speech recognition model with transformer encoders and rnn-t loss. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7829–7833. IEEE.

Xuanyu Zhang. 2019. MCˆ2: Multi-perspective convolutional cube for conversational machine reading comprehension. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6185–6190, Florence, Italy. Association for Computational Linguistics.

Xuanyu Zhang. 2020. Cfgnn: Cross flow graph neural networks for question answering on complex tables. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):9596–9603.

Xuanyu Zhang and Zhichun Wang. 2020. Rception: Wide and deep interaction networks for machine reading comprehension (student abstract). *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(10):13987–13988.

Xuanyu Zhang and Qing Yang. 2021a. Dml: Dynamic multi-granularity learning for bert-based document reranking. In *Proceedings of the 30th ACM International Conference on Information amp; Knowledge Management*, CIKM '21, page 3642–3646, New York, NY, USA. Association for Computing Machinery.

Xuanyu Zhang and Qing Yang. 2021b. Position-augmented transformers with entity-aligned mesh for textvqa. In *Proceedings of the 29th ACM International Conference on Multimedia*, MM '21, page 2519–2528, New York, NY, USA. Association for Computing Machinery.

Xuanyu Zhang, Qing Yang, and Dongliang Xu. 2023. Xuanyuan 2.0: A large chinese financial chat model with hundreds of billions parameters. *arXiv preprint arXiv:2305.12002*.

Guangxiang Zhao, Junyang Lin, Zhiyuan Zhang, Xuancheng Ren, Qi Su, and Xu Sun. 2019. Explicit sparse transformer: Concentrated attention through explicit selection. *arXiv preprint arXiv:1912.11637*.

Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. 2021. Deformable {detr}: Deformable transformers for end-to-end object detection. In *International Conference on Learning Representations*.

Zhenhai Zhu and Radu Soricut. 2021. H-transformer-1D: Fast one-dimensional hierarchical attention for sequences. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3801–3815, Online. Association for Computational Linguistics.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
Left blank.

☑ A2. Did you discuss any potential risks of your work?
Left blank.

☑ A3. Do the abstract and introduction summarize the paper's main claims?
Left blank.

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☒ Did you use or create scientific artifacts?

Left blank.

☐ B1. Did you cite the creators of artifacts you used?
No response.

☐ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
No response.

☐ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
No response.

☐ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
No response.

☐ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
No response.

☐ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
No response.

C ☑ Did you run computational experiments?

Left blank.

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Left blank.

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Left blank.

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
Left blank.

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
Left blank.

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?
Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
No response.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
No response.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
No response.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
No response.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
No response.