ReAT: Long Text Encoder Representations with Attention Tokens

[1]Harunori Kawano, [1]Sota Shimizu

Abstract— **BERT is a Transformer-based language model with a self-attention mechanism and has made many advances in natural language processing. However, it is difficult for BERT to process long texts more than 1,000 words, and it is possible to process them only when reducing the number of sentences. In this paper, a novel language model, ReAT, is proposed and experimented. The authors aim at developing a language model which is as accurate as or more accurate than BERT when processing long texts without reducing the number of sentences. ReAT supports long texts by dividing a given long text into dozens of sentence units as its preprocessing. In order to verify effectiveness of ReAT, we conducted simple experiments using QQP and MRPC after pre-training. As their results, ReAT achieved 89.7 percent accuracy by the QQP test and 73.0 percent accuracy by the MRPC test, respectively, at a stage when pre-training had not been completed. ReAT achieved 32-times longer text processing than BERT (from 512 words to 16384 ones). The amount of computing the self-attention was reduced successfully. In addition, our proposed language model achieved to extract feature based on sentence-by-sentence.**

Ⅰ. Introduction

In recent years, a field of natural language processing in deep learning has made great progress. Techniques in this field have enabled us to process more complex sentences. Transformer [1], proposed in 2017, enables us to execute parallel processing which could not be achieved with conventional RNN-based models [2][3][4]. It solved problems such as vanishing gradients problem in processing long sentences. Further, it has an architecture called self-attention, which has been applied to various fields of machine learning, including image processing and table data processing as well as natural language processing. Bidirectional Encoder Representations from Transformers (BERT) [5] is also a Transformer-based model. Its fine-tuning capability facilitates its application to various tasks such as text classification, summarization, and translation. With the advent of BERT, it was used for various applications in natural language processing, and many BERT-architecture-based models were also proposed [6][7][8][9].

However, BERT has a problem of reduced accuracy when it processes long texts and is not suitable for processing long texts more than 1,000 words. There seem to be two reasons for this problem. The first is that processing a self-attention mechanism (hereafter, Attention) needs to pass through the soft-max layer. Attention computes a matrix product of the query and the key, i.e., Attention weights, and computes another matrix product of the value and Attention weights which passes through the soft-max. In other words, Attention mixes feature vectors between tokens at a certain rate. And as the texts get longer, a proportion of one token becomes smaller, resulting in less information. Intuitively, increasing dimensionality of the vectors and making network layers deeper seems to enable us improve this problem. However, the exponential increment in computational complexity limits this method. The second is that the only one CLS token is defined from an input text in BERT. CLS token is used for tasks such as text classification and is responsible for extracting task-specific feature from some text. Further, visualization of self-attention weights in BERT showed that weights were concentrated into peripheral tokens in lower layers, but as the layers becomes deeper, weights were concentrated into CLS token. This result was discussed in a work by Kevin Clark, and et al. [10]. This indicates that CLS token gathered important features in a text. However, since the only one CLS token was provided for each text, the feature to be aggregated into the CLS token increases as the text becomes longer. In other words, such a feature might prevent the input long text from conducting various tasks of natural language processing accurately. The authors think this problem could be solved by preparing multiple CLS tokens. On the same time, increasing the number of CLS tokens simply might cause the computational complexity. Thus, we need to consider this point of view sufficiently.

Several methods of long texts processing in BERT have been proposed. Head-Tail, proposed by Che San and et al. [11], allowed us to process the long text by extracting the beginning and end of a text. CogLTX, proposed by Ming Ding and et al. [12] identified important sentences and excludes sentences inferred to be unimportant, allowing us to process the long texts. However, all these methods allowed it by just reducing the number of sentences. In other words, it performs very well for tasks which are inferred from parts of a sentence, such as text classification, but is unsuitable for tasks which require the entire text, such as translation.

The purpose of this study is to develop a language model which achieves the same (or better) accuracy as BERT when processing the long texts without reducing the number of sentences. Inaccuracy in the self-attention mechanism in BERT is caused by the length of key, i.e., the length of input text. Therefore, this problem can be solved to some degree by shortening the length of input value (length of input text) of the Attention mechanism. BERT uses the text as input values when extracting text feature, so if a 1000-word text is inputted, the Attention mechanism needs to calculate 1000 tokens. To solve this problem, it seems to be effective to shorten the length of the Attention input vector by dividing the whole text into some sentence units and calculating them.

We cannot use BERT as-is for input values of a text divided

[1] Harunori Kawano, Sota Shimizu are with Department of Design Engineering, Shibaura Institute of Technology, Tokyo, Japan cy19227@shibaura-it.ac.jp, shimizu@sic.shibaura-it.ac.jp

2023 IEEE/SICE International Symposium on System Integration (SII) | 979-8-3503-9868-7/23/$31.00 ©2023 IEEE | DOI: 10.1109/SII55687.2023.10039230

Authorized licensed use limited to: University of Maryland College Park. Downloaded on March 23,2023 at 01:12:01 UTC from IEEE Xplore. Restrictions apply.

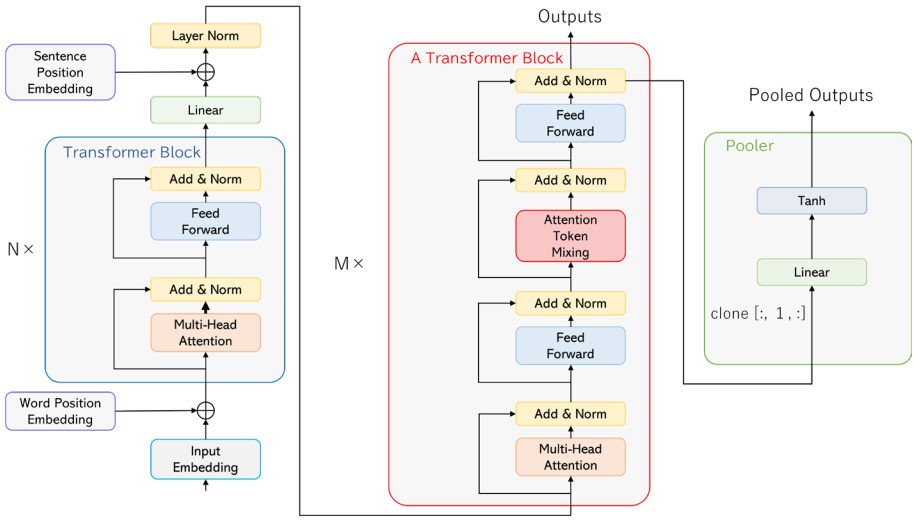

Fig. 1. Overall view of ReAT model, composed of Embedding, an N-layer Transformer layer, and an M-layer Alpha Transformer layer.
(In the base model, the Transformer Block consists of 3 layers and the Alpha Transformer Block consists of 4 layers.)

by sentences. This is because BERT assumes that the whole text is inputted as it is, and it is impossible to consider the feature between sentences when the text is divided into some sentence units. To solve this problem, we propose a novel language model called Long Text Encoder **Re**presentations with **A**ttention **T**okens (ReAT). ReAT is based on BERT architecture, but our own defined token called Attention token is added. Attention Token Mixing is incorporated into the architecture to enable us to extract a context feature between the sentences. In addition, ReAT adds Single Sentence token (hereafter, SS token) to place of CLS token, enabling us to extract feature based on a sentence-by-sentence through pre-training. Chapter I of this paper mentioned our research background, purpose and motivations. Chapter II proposes our ReAT model. In addition, its details are explained. Chapter III experiments our ReAT model. Chapter IV discusses experimental results. Chapter V summarizes this paper.

II. Model

This chapter introduces ReAT and explains its implementation in detail. ReAT is a language model optimized for processing long texts and requires two-step training: pre-training and fine-tuning in the same way as BERT [5]. In the following, we describe the preprocessing, the structure, and the calculation process of ReAT model with specific examples.

A. Preprocessing

First, the input text is divided by sentence breaks such as '.', '!', '?', and etc. The number of dimensions increases as a size of array. In other words, a text is transformed as [sentence 1, sentence 2, ..., sentence *N*]. Next, words are converted to IDs through a tokenize process. A tokenizer is the same as that in uncased BERT. ReAT prepares a SS token (<SS>), an Attention token (<Attn>), and an EOS token (</s>) as special tokens in the tokenize process. They are added to the first word, the second word, and the end of the sentence, respectively. SS token has a role of extracting feature of a single sentence, while Attention token has a bridging role of extracting context feature between sentences. SS tokens are used for classification tasks, and etc. Attention tokens are discarded when encoding of the text is completed. The final input value to ReAT model is a two-dimensional array composed of "Number of sentences" and "Max sentence length".

B. Model Architecture

ReAT is based on BERT and has a stacked structure of Embedding, Transformer layers and Alpha Transformer layers. Figure 1 shows the overall view of ReAT. For the first input value, Word Embedding and Word Position Embedding are performed. In BERT, the Position ID is assigned from the first word of a text, but in ReAT, the Position ID is assigned from the first word of a sentence, since the division is defined in units of sentences. After the initial Embedding process, feature extraction is performed in Transformer layers. Transformer layers consist of Multi-head Attention and Feedforward as in BERT. Transformer layer extracts feature based on a sentence-by-sentence and does not take into consideration context feature between sentences. After Transformer layers, output values are transmitted to Affine, Sentence Position Embedding, and Alpha Transformer layers, in this order. The Affine layer is responsible for increasing the hidden size, which is intended to improve the accuracy of context feature extraction between sentences. Sentence Position Embedding is responsible for embedding the position of a segmented sentence as a feature. Alpha Transformer layers consist of Multi-head Attention and Feedforward, Attention Token Mixing to extract feature in text units. Pooler is responsible for processing CLS tokens in BERT, such as pooling them according to task, but in ReAT, Pooler performs the same processing for SS tokens.

Authorized licensed use limited to: University of Maryland College Park. Downloaded on March 23,2023 at 01:12:01 UTC from IEEE Xplore. Restrictions apply.

C. Attention Token Mixing (ATM)

ATM uses Attention tokens to extract context feature between sentences. The structure of Multi-head Attention is the same as the conventional one, but the input value differs from the normal self-attention. In ATM, as shown in Fig.2, all tokens are inputted into query, and only the Attention tokens are inputted into the key and the value. This calculates Attention between Attention tokens and all tokens and performs context feature extraction between sentences. Attention mask in ATM is a binary matrix which identifies the padding positions of sentences.

D. Pre-training

Two types of tasks were prepared for pre-training of ReAT. The first is Masked Language Model (MLM), in the same way as BERT, and the second is Exchange Sentence Prediction (ESP), newly-developed in this study. MLM is a task which masks words in a sentence at a certain rate and infers the masked words from the entire sentence. ESP is a task for training SS tokens efficiently. Although this paper does not test the effectiveness of ESP in pre-training, we think that training MLM and ESP simultaneously could facilitate its application to various tasks during fine-tuning.

E. Exchange Sentence Prediction (ESP)

ESP is a task which swaps the position of sentences at a certain rate and infers from the output of SS token whether the sentences were swapped. SS token is a token which is placed at the beginning of each sentence and is responsible for extracting the feature of a sentence. However, it seems to be difficult to extract sentence-specific feature using only MLM, because the features of Attention tokens from other sentences are mixed into the input sentence. ESP helps us to extract sentence-specific feature because each sentence-specific value is the correct label.

Ⅲ. Experiments & Results

In this paper, we denote the number of Transformer blocks as L, the number of Alpha Transformer blocks as AL, the Transformer blocks hidden size as H, and the Alpha Transformer blocks hidden size as AH. In experiments, we used ReAT-base (L=3, AL=4, H=768, AH=1024, Parameters =138M), i.e., the base model of ReAT, and ReAT-small (L=2, AL=2, H=512, AH=768, Parameters=51M), i.e., the small model of ReAT, for pre-training and simple benchmark tests. Quora Question Pairs (QQP) and Microsoft Research Paraphrase Corpus (MRPC) of General Language Understanding Evaluation (GLUE) [13] were used for benchmark tests.

A. Pre-training

The training data was English of C4 dataset [14], and the MLM mask rate was set to 15 percent and the ESP exchange rate to 20 percent. 90 percent of the training was done with the maximum of 48 words each sentence and 16 maximum sentences. The remaining 10 percent was done with the maximum of 128 words each sentence and 128 maximum sentences. An Nvidia V100, RTX3060 was used for training and was able to finish training to 2 percent for ReAT-base and 10 percent for ReAT-small compared to BERT [5]. Detailed training hyperparameters are shown in Table 1.

```
def a_transformer_block(inputs, attn_mask, sen_attn_mask):
    shortcut = inputs
    x = multi_head_attention(
        query=inputs, key=inputs, value=inputs, mask=attn_mask
    )
    x = norm(x + shortcut)

    shortcut = x
    x = feed_forward(x)
    x = norm(x + shortcut)

    shortcut = x
    attn_tokens = x[ :, 1, :]
    x = x.view(seq_num * seq_len, hidden_size)
    x = multi_head_attention(
        query=x, key=attn_tokens, value=attn_tokens, mask=sen_attn_mask
    )
    x = x.view(seq_num, seq_len, hidden_size)
    x = norm(x + shortcut)

    shortcut = x
    x = feed_forward(x)
    out = norm(x + shortcut)

    return out
```

Fig. 2. Alpha Transformer layer program: Inputs are an array of the shape (sentences num, words num, hidden size).

B. GLUE

Since ReAT has the same processing as BERT when there is only one sentence, experiments were conducted using QQP and MRPC with more than one sentence in order to verify the effectiveness of ReAT. QQP and MRPC are tasks which take two sentences and perform binary classification. In training these tasks, ReAT uses Pooled output. Since ReAT inputs sentences in a two-dimensional array, the two input sentences are transformed into a two-dimensional array ([[sentence 1], [sentence 2]]) when training them. Since the Pooled output is (2, hidden size), it was transformed to (hidden size × 2) and passed through the Affine layer for binary classification. As a result, ReAT-base achieved 89.7-percent accuracy on the QQP test and 73.0-percent accuracy on the MRPC test. Verification results are shown in Table 2.

Ⅳ. Discussion

As results of the above validation experiments, ReAT succeeded in producing a certain level of accuracy in MLM, and its effectiveness as a language model could be confirmed. Benchmark tests using QQP and MRPC showed differences in accuracy when comparing their results. ReAT achieved 89.7-percent accuracy in the QQP test, only one point lower than BERT [5]. However, the MRPC test did not yield good results necessarily as an accuracy of 73.0 percent. This result is attributed to the fact that the amount of data for MRPC was considerably smaller than that for QQP and that the pre-training had not been completed. Therefore, we can expect to achieve results equal to or better than BERT at the end of the pre-training phase.

The validation experiments in this paper could not achieve

Authorized licensed use limited to: University of Maryland College Park. Downloaded on March 23,2023 at 01:12:01 UTC from IEEE Xplore. Restrictions apply.

Table1: pre-train hyper-parameters.

Hyperparameter	ReAT-base	ReAT-small
Number of Transformer layers	3	2
Number of Alpha Transformer layers	4	2
Transformer Hidden Size	768	512
Alpha Transformer Hidden Size	1024	768
Transformer FFN Hidden Size	3072	2048
Alpha Transformer FFN Hidden Size	4096	3072
Optimizer	AdamW (β_1=0.9, β_2=0.999, weight decay=0.01, ε =1e-6)	
Learning rate	4e-5	2e-4
Scheduler	10000 step warmup & linear decay	
Dropout	0.1	0.1
Batch size (number of sentences)	256(avg. 18 texts)	512(avg. 36 texts)
Train steps	25M	5M

the final verification of the effectiveness of ReAT and Attention token. This paper planned to implement a machine translation model and verify the effectiveness of ReAT in processing long texts. However, due to inadequacies in the development environment, it was not possible to proceed beyond the benchmark test using QQP and MRPC. Verification continues when the development environment is ready. Further, ReAT succeeded in producing a certain level of accuracy with ESP, confirming the effectiveness of Attention token. However, the limitations of Attention token could not be verified, and this also needs to be verified once the pre-training is completed.

ReAT has three advantages beyond BERT as itemed in the following.

1. The maximum number of words which can be processed is higher than that of BERT. ReAT can process up to 16384 words (128 words x 128 sentences), which is 32 times more words than BERT.
2. The computational complexity of Attention has been reduced. The computational complexity of Attention weight is proportional to square of the length of the input value (sentence length). The length of the Attention input value was reduced because ReAT processes the text in multiple sentence units. As a result, the computational complexity, which was $O(L^2)$ in BERT, was reduced to $O(L^2/N)$ in ReAT (L=length of text, N=Number of sentence segments).
3. Sentence-by-sentence feature extraction is possible. BERT has a CLS token for each text, while ReAT has a SS token for each sentence. ReAT obtains the features of each sentence from the SS token by pre-training. This allows features which could not be obtained from CLS tokens to be obtained from SS tokens.

ReAT gained an advantage beyond BERT in three ways. However, the number of parameters and paddings increased. The number of parameters for BERT-base was 110M, while the number of parameters for ReAT-base was 138M. In addition, the number of paddings was largely increased by dividing a text into multiple sentence units. At the stage of converting a variable-length sentence to a fixed-length one, ReAT had to standardize the number of sentences in the text, so an array with only padding was generated.

ReAT can process 32 times as many words as BERT. However, the Attention structure itself has not changed, which limits the length of text which can be processed. Some suitable mechanisms must be incorporated to solve the problems fundamentally in processing long texts, and this becomes a more important issue in the future. Further, the longer the text to be processed, the larger the number of paddings. Enabling us to process variable-length sentences is an important issue in natural language processing.

Ⅴ. Conclusion

BERT [5] is a language model which incorporates self-attention and has performed very well compared to traditional RNN-based models. However, there is a problem of reduced accuracy in processing long texts more than 1,000 words. Several methods have been proposed for processing long texts, but they enabled us to process long texts by reducing sentences and were not suitable for tasks, which use entire text, such as translation.

The purpose of this paper is to develop a language model which is as accurate as or more accurate than BERT in processing long texts without reducing the number of sentences. We added sentence-wise splitting of the long text to the preprocessing and proposed a novel language model called ReAT, which can process text in two dimensions. ReAT has a context feature called Attention token, which

Table2: task accuracy.

Accuracy	ReAT-base (500K steps)	ReAT-small (500K steps)	BERT-base
MLM	61.8	60.8	
ESP	79.7	79.3	
QQP	89.7	89.0	90.8
MRPC	73.0	75.0	88.9

Authorized licensed use limited to: University of Maryland College Park. Downloaded on March 23,2023 at 01:12:01 UTC from IEEE Xplore. Restrictions apply.

incorporates a mechanism to extract features between sentences. Further, the addition of a SS token in place of the CLS token has made it possible to extract features based on sentence-by-sentence.

ReAT is a two-step training model (pre-training and fine-tuning) as well as BERT, so a pre-training was conducted at the beginning. Simple experiments were then conducted using QQP and MRPC of GLUE [13]. Due to inadequacies in the development environment, ReAT cannot conduct the same pre-training as BERT. In spite of such a condition, QQP test showed a good result, but MRPC did not necessarily. However, it became possible to process up to 16384 words of text at a time, and the amount of weight calculation in Attention was successfully reduced from $O(L^2)$ to $O(L^2/N)$ (L=text length, N=number of divisions of text).

Although this paper increased the number of words which could be processed compared to BERT, it did not solve the problem fundamentally. It is necessary to incorporate another novel mechanism to fundamentally solve the problem which is caused by the number of words which can be processed by the language model. Further, enabling us to process variable-length sentences in natural language processing is a very important task obviously.

References

[1] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin, Attention is all you need, Advances in Neural Information Processing Systems, pp. 6000–6010 (2017).

[2] T. Mikolov et al., "Recurrent neural network based language model", Proc. of 11 th Annual Conference of the International Speech Communication Association (INTERSPEECH 2010), pp. 1045-1048 (2010).

[3] Sutskever, I., Vinyals, O., and Le, Q., Sequence to sequence learning with neural networks, Advances in Neural Information Processing Systems (NIPS 2014), (2014).

[4] Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer, Deep Contextualized Word Representations, Proc. of NAACL, (2018).

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

[6] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar S. Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke S. Zettlemoyer, and Veselin Stoyanov, Roberta: A robustly optimized bert pretraining approach. ArXiv, abs/1907.11692, (2019).

[7] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter, arXiv preprint arXiv:1910.01108, (2019).

[8] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu, Exploring the limits of transfer learning with a unified text-to-text transformer, arXiv preprint arXiv:1910.10683, (2019).

[9] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut, Albert: A lite bert for self-supervised learning of language representations, arXiv preprint arXiv:1909.11942, (2019).

[10] Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning, What does bert look at? an analysis of bert's attention, arXiv preprint, abs/1906.04341, (2019).

[11] Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang, How to fine-tune BERT for text classification?, Proc. of CCL, (2019).

[12] Ming Ding, Chang Zhou, Hongxia Yang, and Jie Tang, CogLTX: Applying BERT to Long Texts, Advances in Neural Information Processing Systems 33 (2020).

[13] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman, Glue: A multi-task benchmark and analysis platform for natural language understanding, Proc. of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP, pp. 353–355 (2018).

[14] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu, Exploring the limits of transfer learning with a unified text-to-text transformer, Journal of Machine Learning Research, (2020).

Authorized licensed use limited to: University of Maryland College Park. Downloaded on March 23,2023 at 01:12:01 UTC from IEEE Xplore. Restrictions apply.