BERT Pre-training Acceleration Algorithm Based on MASK Mechanism

Puzhao Hu, Lanfang Dong and Yanyan Zhan
University of Science and Technology of China, Hefei, Anhui, China
Email: 1061788407@qq.com; lfdong@ustc.edu.cn

Abstract. With the development of natural language processing (NLP) science, machines can understand human language better and better. Especially with the emergence of pre-training models in recent years, such as ELMO, GPT, BERT, etc. Researchers can easily transfer model knowledge and apply them to related fields, but at the same time they often ignore the pre-training process. Due to the huge number of parameters and calculations of the pre-training model, it often takes a lot of resources and much time to get a good pre-training model. So many people are daunted by the pre-training process. From a different perspective, this article mainly focuses on the pre-training process of BERT, and conducts in-depth research and related improvements on its own mask mechanism, so as to accelerate the training process. The experimental results show that our algorithm can get better result than the original scheme which the BERT use. Also it is better than the original BERT in terms of training speed and can better understand the training text input. What is more, we have also conducted a detailed analysis of the experimental result.

Keyword. Natural language processing; pre-training; BERT; mask.

1. Introduction
At present, the development process of natural language processing has gone through three main stages, including the first stage represented by statistical machine learning, the second stage represented by word2vec [1], and the third stage represented by BERT [2]. At first, in order to express human language in a way that computers can understand, earlier researchers invented many methods, such as using one-hot encoding to represent a word, bag-of-words to represent a paragraph of text, and uses TF-IDF frequency to express the importance of a word and so on. However, these statistical-based language model has big problems. For example, as the scale of the problem grows, the computational complexity will increase exponentially, resulting in the statistical-based language model cannot model the long-term dependence of the language in the context.

Due to the existence of these shortcomings, in 2003 Bengio first incorporated the idea of deep learning into the language model in his classic paper [3], and proposed the first language model NNLM (Neural Net Language Model), in which the author is very creative, by using the first-level feature mapping matrix of the model as a distributed representation of words, so that a word can be represented in a vector form. Based on this inspiration, Tomas Mikolov published several epoch-making works in 2013, and proposed the CBOW and Skip-gram models for the first time [1]. The emergence of word2vec has greatly promoted the development of NLP, especially the application of deep learning in NLP. Since then, using pre-training word vectors to initialize the first layer of the network structure has become a standard configuration.
On the basis of the predecessors, a large number of pre-training language models began to appear, because when dealing with specific language tasks, if you continue to let the machine learn language representation and understanding from scratch, it often just repeats the work of the predecessors and wastes a lot of time and resources. Pre-training language models are often trained through a huge corpus to make the model truly understands the human language. On this basis, when dealing with a specific task, researchers only need to fine-tune the model according to the characteristics of the task, so that the model can better grasps the knowledge of the corresponding task. Based on this idea, a large number of pre-training language models began to emerge, such as the ELMo (Deep Contextualized Word Representations) network that uses two-way LSTM for feature extraction [4], the one-way model GPT which uses transformer’s decoder process for feature extraction [5], and the two-way pre-training model BERT that adopts the same idea [2]. The emergence of these pre-training models has greatly promoted the development of natural language processing disciplines. Among them, BERT has brought epoch-making changes to the development of NLP with its superior results in various tasks, which is also the focus of this article.

2. Related Work

2.1. Overview
Language model pre-training has been shown to be effective in improving many natural language processing tasks. These include sentence-level tasks, such as natural language inference [6], which aim to predict the relationships between sentences through overall analysis, as well as the token-level tasks such as named entity recognition and question answering, where models are required to produce fine-grained output at the token level.

Currently, there are mainly two existing strategies for applying pre-training language representations to downstream tasks: feature-based and fine-tuning. Feature-based methods, such as ELMo [4], use task-specific architectures that include pre-training representations as additional features. Fine-tuning-based methods such as Generative Pr-trained Transformer introduce the smallest task-specific parameters, and train downstream tasks by simply fine-tuning all pre-training parameters [5]. The two approaches share the same objective function during pre-training, where they use unidirectional language models to learn general language representations. In the same year, the BERT model proposed by the Google AI team has produced epoch-making changes in both the pre-training language model and the development of natural language processing. The structure of the is shown in figure 1. As we can see, the model structure of BERT and GPT are almost the same, and they are all formed by partially stacking the encoder in the transformer, but unlike GPT, BERT proposes a two-way concept. At the same time, the BERT is much larger, and the pre-training corpus of it is several times larger than the GPT model. For many natural language processing tasks, BERT also brings significant performance improvements.

The BERT model consists of an embedding layer, an encoder chain and an output layer. The input word is first embedded as a vector, and then converted by the pipeline of the encoder, and the final prediction is obtained in the output layer. This model is known to require a lot of calculations, leading to higher infrastructure requirements and latency. However lower latency is essential for a good customer experience, therefore, it is very important to design a method that can reduce the computing requirements of BERT and accelerate BERT to successfully meet the delay and resource requirements of the production environment. Recent research has focused on optimizing two basic indicators: model size and inference time. The recently proposed ALBERT achieved significant compression of BERT by sharing parameters between encoders and decomposing the embedding layer [7]. However, since the amount of calculation during the inference period remains the same (even if the training speed is faster), it has almost no effect on the inference time. Other research aims to optimize both indicators at the same time. Here, the natural strategy is to reduce the number of encoders, and DistilBERT has adopted this idea [8]. In the knowledge distillation paradigm, another method is to shrink a single encoder. At the same time, in order to reduce the size of the model, some researchers have also
proposed the ability to learn a large model through a small model. For example, TinyBERT uses a similar method [9]. But generally speaking, obtaining a small model through distillation has an improved algorithm effect compared to the original small model, but it cannot fully guarantee the effect of the large model. General models use FP32, that is, single-precision floating-point numbers for storage and calculation. In response to this problem, some researchers have proposed a mixed-precision acceleration scheme, which is to replace the FP32 variable in the model with FP16, which can effectively compress the size of the model, while improving the training and reasoning speed of the model, it will not affect the effect of the model.

2.2. Our Work
At present, Google’s AI team has open sourced the pre-training BERT model on massive data, including Chinese pre-training model and English pre-training model. When using BERT for a specific task, it is often only necessary to use the pre-training model and just fine-tune it. Because of the huge number of parameters of the BERT model, taking BERT-base as an example, there are about 110 million parameters. The pre-training process takes a lot of time and resources, so few researchers pay attention to it. However, in this article, we mainly focus on the pre-training process of BERT itself, analyze its own mask mechanism, point out its unreasonable places and make corresponding improvements. For each improvement scheme, we conducted corresponding experiments by simulating the pre-training task of BERT itself, namely the masking language model, and analyzed the feasibility of the scheme proposed in this paper through the analysis of the experimental results.

3. Algorithm
This section is organized as follows. Section 3.1 introduce the BERT structure and pre-training tasks, the mask mechanism used by BERT and the improved algorithm we proposed in Section 3.2.

3.1. BERT Pre-training Method
This chapter first briefly introduces the model structure of BERT and its training process. As mentioned above, BERT and GPT are the same in structure, and both are formed by stacking the encoder layers of the transformer. At the same time, BERT uses an unsupervised method for training, on a corpus of 3.3 billion words, without any annotation data. At the same time, BERT proposed a bidirectional (Bidirectional) concept, which is different from the unidirectionality of the GPT. When predicting a word, BERT will combine the information before and after the word to be predicted, as shown in figure 1 below.

![Figure 1. Model structure of BERT and GPT.](image)

BERT pre-training includes two tasks. The first task is to randomly deduct 15% of the words and replace them with a mask to make the model guess the word. It is about to input the masked sentence into the BERT, and classify the position vector corresponding to the mask in the model output matrix and the label is the subscript corresponding to the masked word in the dictionary. The specific process is shown in figure 2. The second task is that each training sample is a upper and lower sentence, with
50% for the sample, the next sentence and the previous sentence are true. For the other 50% of the samples, the next sentence and the previous sentence are irrelevant. The model needs to judge the relationship between the two sentences. In the prediction, the corresponding vector of [CLS] is taken out for two-class training.

Figure 2. Bert pre-training flowchart.

Figure 3. Model structure.

3.2. Our Algorithm
Knowing the pre-training process of BERT, we can easily know that one of the biggest problems in the model pre-training process is that the convergence speed of BERT will be slow. As we know that the pre-training process of BERT is similar to cloze filling, and can only predict the data that is masked each time. However, because only 15% of the words in each batch of data are removed by mask, this makes BERT can only predict 15% of the input words at a time, causing the model to require more steps to converge. Though use the mask mechanism to predict the results by contacting the context information through attention during training, which can better learn semantic representation, but it is obviously an unreasonable way to predict only 15% of the words each time. In response to this problem, this paper has carried out some exploratory experiments, and the experimental results have also verified the feasibility of our improvement plan.

Our improvement plan mainly focuses on the mask mechanism of BERT itself, and does not change the model structure of BERT. Aiming at the problem of slow training convergence caused by the mask mechanism of BERT, this article proposes the following tentative improvement plans:

1) Since BERT only mask 15% of the information in the original sentence each time, in which 80% is replaced with MASK, 10% is replaced with any word, and 10% remains unchanged. An obvious idea is that if more words are masked, more words in the original sentence can be predicted each time, and then the convergence speed of the model can be accelerated. Based on this idea, we set the mask rate to 15%, 35%, and 50% respectively for comparative analysis.

2) Refer to GPT’s mask mechanism, when predicting the current word, the GPT will mask all the following information at the same time, but because it uses the full amount of tags, it can predict all the words of the sentence for each input. By comparing the mask mechanism of BERT (as shown in figure 3), we can intuitively see the inefficiency of BERT. Benefiting from the inspiration of the GPT model, we proposed three improvements (triangle mask, diagonal mask, around mask) for the mask mechanism of BERT itself, as shown in figure 4.

(2.1) The diagonal mask: As show in figure 4, referring to the mask mechanism of GPT, here we mask the current input. At the same time, in order to ensure the bidirectionality of the BERT itself, we do not mask the words behind the current word, so that the prediction of the current word can also be based on the information from the previous and subsequent texts.
(2.2) The triangle mask: GPT cannot predict the first word of the input, because at this time, it does not get any previous information. Based on this idea, we have improved the first scheme, which is also shown in figure 4.

(2.3) The around mask: The above schemes predict the previous word or the next word of the current word, so can it predict multiple words at the same time? In response to this idea, we propose the around mask. As shown in figure 4, for the current word, it will predict the words before and behind it at the same time.

![Figure 4. Our proposed mask mechanism.](image)

4. Experiments

4.1. Experimental Setting
Model: Limited by the hardware resources, we use the 4-layer BERT model in this paper, that is, the main part is mainly formed by a stack of 4-layer transformer encoders, and the number of attention-heads we use is 6. The model structure is shown in figure 3 above and parameters we choose is in table 1.

| N-layer | Input-Dim | Attention-head | Learning-Rate | Drop-Rate | Batch-Size |
|---------|-----------|----------------|---------------|-----------|------------|
| 4       | 256       | 6              | 0.0001        | 0.25      | 64         |

Date: For the experimental data set, we use the official Microsoft data set MRPC, which mainly includes 5800 sentence pairs extracted from web news. Since this article only explores the first training task of BERT, that is, the masked language model (MLM), we have removed the 0,1 label that represents the context relationship before the sentence pair.

4.2. Result and Analysis
We conducted corresponding experiments for each of the proposed scheme, and compared the experimental results with the original BERT solution. The results are as follows:
Firstly, we gradually increase the probability P of the mask rate, we will train the model for 5000 steps when we take different values for each P. For the input “<GO> problematic has a margin of error of plus commuters minus <NUM> percentage points. <SEP> that poll had <NUM> likely voters and sampling error of one-third or thirty-four <NUM> percentage”, The prediction results under different mask rates are shown in the following table 2. As we can see, when the probability P of the mask is gradually increased, the convergence speed of the model will gradually become faster, but when the mask probability is too large, the convergence speed will become slower. As shown in table 2 below, when we set the mask probability to 0.30, when we train 5000 steps, the loss at this time is less than the case of p=0.15 which is used by the original model. At the same time, the prediction result of the mask word is also better than the case of p=0.15. However, when you continue to increase p to 0.50, the loss at this time will increase instead, and the predictions are all invalid words, which means that the model does not learn useful information at this time. Our explanation for this situation is that when the probability P of the mask is increased, more words can be learned by the model for each training, which can speed up the training convergence speed of the model. However, when P is too large, in the specific prediction of a word, because a large number of words in the context are masked, it is more difficult for the model to obtain contextual information, and it is also difficult for the model to learn the understanding of semantics, so on the contrary, the convergence speed will slow down, and it cannot speed up the training speed of the model.

| P    | Prediction                                                                 | Loss  |
|------|----------------------------------------------------------------------------|-------|
| 0.15 | target:it, or, plus, minus Predict:it, and, <NUM>, minus                   | 5.071 |
| 0.30 | target:<GO>, a, it, computers, or, plus, minus, sampling predict:<GO>, a, do, those, or, ‘’, the, people, percentage | 4.275 |
| 0.50 | target:<GO>, has, a, of, computers, minus, <NUM>, points, that, poll, likely, voters, and, of predict:<GO>, a, a, ‘’, ‘’, ‘’, <NUM>, the, ‘’, the’, ‘’, ‘’, ‘’, the, of | 5.441 |

(2) When modifying the MASK mechanism of BERT itself, for each improvement scheme we proposed, we record all the loss results at 2000 steps, 3500 steps, and 5000 steps respectively (table 3). Also for the same input “Mrs clinton said her husband action’s were morally wrong but did not constitute a public bet”, we record the prediction results of each scheme after training for 5000 steps (table 4).

It can be seen from the experimental results that whether it is the triangle_ mask or the around_mask algorithm, comparing to the mask mechanism of the original model, under the same iterative training 5000 steps, the loss function we propose is much smaller than BERT itself. The experimental results shows that our algorithm can indeed speed up the convergence of BERT training. The reason is easy to understand, unlike BERT, which can only predict 15% of the input per training, our solution uses a fully quantized mask label, so all input data can be predicted each time. Similarly, as shown in table 4, after training for 5000 steps, our scheme can make more accurate predictions on the input compared to the original BERT scheme.
Table 4. Prediction results after 5000 steps.

| Method       | Predict                                                                 |
|--------------|-------------------------------------------------------------------------|
| original_mask| mask:{clinton,morally} predict{the,really} Mrs Clinton said her husband action’s were morally wrong but did not constitute a public bet |
| triangle_mask| Mrs Clinton said her husband action’s were morally wrong but did was want a public bet |
| around_mask  | Mrs Clinton said her husband action’s were morally wrong but did not made a public bet |
| diagonal_mask| Mrs Clinton said her husband action’s were morally wrong but did not constitute a public bet |

At the same time, we visualized the results of the first attention-head and compared them with the visualization result of the original model as shown in figures 5 and 6. Through the attention block diagram and line graph, it is not difficult to find that the mask scheme(triangle_mask, around_mask) we proposed will not change the bidirectionality of BERT itself, besides compared with the original model, in the case of training the same number of steps(5000), the attention line graph will have more line segments crossing, which means that when predicting the current word, the model can better consider the previous and subsequent information of the input, which means that it can better understands the input information and the training text.

In this paper, we do not give the attention visualization result when using the diagonal mask mechanism. Because in comparison with other mask mechanisms (table 3), it is not difficult to find that when using the diagonal mask mechanism, the model can quickly converge. It shows amazing performance when training for about 2000 steps, but this is actually unreasonable. Through the analysis of its internal mechanism, we found that the mask mechanism itself is wrong, because when the model predicts the masked word, the model itself can see the original word because of the existence of the residual structure, so even if the model can quickly converge, but in fact, no useful information is learned, because the model knows the input indeed, it just output the result according to the input when predicting, which is meaningless.

In terms of convergence speed, the around_mask and triangle_mask algorithm are similar. But after the same training for 5000 steps, the_around mask algorithm is obviously better (table 4) than the original model and also better than the triangle_mask algorithm. In predicting a certain word, the around mask algorithm can get better attention (figures 5 and 6), indicating that it has a better understanding of the training text and can better learn the semantic features expressed in the text.

Figure 5. Attention mechanism block diagram.
5. Conclusion
Compared with other works for compressing the size of BERT itself, this article explores BERT acceleration method from another perspective, that is, accelerating its own pre-training process. In this article, we conducted a comparative experiment on the probability of the mask in the BERT pre-training process, and analyzed the pros and cons of the model pre-training performance under different mask probabilities. At the same time, in view of the problem that original mask mechanism of BERT can only predict a small part of the input words each time, we propose three tentative improvements (triangle mask, diagonal mask and around mask). The experimental results prove that when we use the full tag of mask, under the same number of training steps, whether it is a triangle mask or around mask algorithm, it can better learn about the input text, and it can also converge faster. It shows that our solution is much better than the original mask mechanism both in semantic learning and training speed.

Acknowledgment
This research is sponsored by the key special project of the National Key R&D Program “Intelligent Robots”, the project number is 2020YFB1313600, and the name is “Theories and Methods of Social Interaction Driven by Robot Needs and Emotions”.

References
[1] Tomas M, Chen K and Greg C 2013 Efficient Estimation of word representations in vector space International Conference on Learning Representations p 311.
[2] Devlin J, Chang M W and Lee K 2018 BERT: Pre-training of deep bidirectional transformers for language understanding North American Chapter of the Association for Computational Linguistics p 763.
[3] Yoshua B, Ré Jean D and Christian J 2003 A neural probabilistic language model Journal of Machine Learning Research 1137-1155.
[4] Peters M, Neumann M and Iyyer M 2018 Deep contextualized word representations *North American Chapter of the Association for Computational Linguistics* p 472.

[5] Alec R, Karthik N, Tim S and Ilya S 2018 *Improving Language Understanding with Unsupervised Learning* Technical Report OpenAI.

[6] Samuel R B, Gabor A, Christopher P and Christopher D 2015 A large annotated corpus for learning natural language inference *Empirical Methods in Natural Language Processing* p 127.

[7] Lan Z Z, Chen M D and Gimpel K 2020 *A lite BERT* for self-supervised learning of language representations *International Conference on Learning Representations*.

[8] Victor S, Lysandre D and Wolf T 2019 *DistilBERT*, a distilled version of BERT: smaller, faster, cheaper and lighter arXiv: *Computation and Language* p 985.

[9] Jiao X Q, Yin Y C and Shang L F 2020 *TinyBERT*: Distilling BERT for natural language understanding *Empirical Methods in Natural Language Processing* (2020) 229.