Creating a very small BERT pretrained model Main Quest 04

Anonymous authors
Paper under double-blind review

Abstract

The rapid evolution of Natural Language Processing (NLP) has been greatly influenced by the advent of the Transformer architecture and the subsequent rise of transfer learning methods, prominently featuring pre-training and fine-tuning phases. This research delves into BERT (Bidirectional Encoder Representations from Transformers), a pivotal model in the realm of NLP, to dissect its capabilities and limitations. Our study has two primary contributions. First, we construct a downscaled version of a BERT pre-trained model to unravel its inner mechanics. Second, we assess the small-scale BERT model's efficacy in understanding complex linguistic patterns. Despite leveraging computational efficiency techniques like np.memmap and training on a smaller 1/7th subset of our 1.4GB dataset, our miniaturized model demonstrated suboptimal performance with accuracy rates around 0.6. This reiterates the importance of optimized, larger-scale BERT variants like ALBERT and DistilBERT for maintaining high performance. The paper also highlights challenges related to fine-tuning BERT-Large models, such as model instability and significant performance variances, thereby outlining areas for future research to make BERT-based models more robust and reliable.

1 Introduction

The field of Natural Language Processing (NLP) has undergone significant transformation with the introduction of the Transformer architecture by Google in 2017, which subsequently enabled the advent of transfer learning. Transfer learning typically consists of two phases: pre-training and fine-tuning. In the pre-training phase, a model is trained on a large corpus to gain a general understanding of language, and this pre-trained model is then fine-tuned for specific NLP tasks. This approach has yielded astonishing improvements in performance across a variety of NLP tasks. Despite the significant advancements in this area, there is an increasing need to understand these models better and make them computationally efficient. The research presented here aims to contribute to this discourse by focusing on BERT (Bidirectional Encoder Representations from Transformers), a groundbreaking model that has shown remarkable success in capturing the intricate semantics of language.

The contributions of this paper are twofold. Firstly, it aims to enhance the understanding of BERT's inner workings by constructing a very small version of the pretrained model. Secondly, it evaluates the capabilities and limitations of this small-scale BERT model in capturing complex linguistic features. The paper is organized into sections that cover background information on unidirectional drawbacks of models like ELMo and GPT, the methodology involving BERT's architecture and pretraining phases, and a conclusion highlighting the merits and demerits of BERT, with a particular focus on the challenges of fine-tuning.

2 Background

2.1 Transformer and Transfer Learning in NLP

The introduction of the Transformer architecture enabled more effective techniques for pre-training language models, thereby paving the way for transfer learning in NLP. Notable models that utilize the Transformer

architecture for transfer learning include GPT, which is trained with a Left-To-Right (LTR) structure, and ELMo, which emphasizes contextual features.

2.2 Feature-Based vs Fine-Tuning Approaches

There are two primary techniques for leveraging pre-trained language models: feature-based learning and fine-tuning. ELMo exemplifies feature-based learning by employing two LSTM language models trained in LTR and Right-To-Left (RTL) directions. The hidden states from each layer are combined to offer a contextual representation of words in a sentence. This approach addresses a significant limitation in traditional word2vec embeddings, which do not consider the contextual meaning of words. In contrast, GPT follows a fine-tuning approach by adjusting all the pre-trained model parameters for specific downstream tasks.

2.3 Unidirectional Language Models and Their Limitations

BERT has highlighted a crucial drawback with unidirectional models, particularly those built on an LTR structure, like GPT. Such models often fail to capture context effectively, especially in tasks requiring bidirectional understanding, such as Question Answering tasks. BERT addresses this limitation by introducing a new Masked Language Model, which aims to capture richer contextual information during the pre-training phase. While ELMo also attempts to incorporate bidirectional information, it does so by simply combining the outputs of individually trained LTR and RTL models. This approach falls short in capturing deeper contextual relationships among words in a sentence.

3 Method

The BERT model used in the research has 3 Transformer layers, a hidden size of 256, and 4 multi-head attentions. In the "Method" section, we will delve into the architecture of BERT, focusing on its unique pretraining techniques that allow it to capture bidirectional context more effectively than its predecessors. BERT is trained in two distinct phases: Pre-training and Fine-tuning. In the Pre-training phase, the model performs two types of tasks on unlabeled data. Then, in the Fine-tuning phase, the pre-trained parameters are loaded and updated to fit a specific labeled task. Therefore, even though the same pre-trained model is used, it undergoes fine-tuning to adapt to the problem at hand, resulting in a specialized model.

3.1 Pre-training

3.1.1 Masked Language Model

MLM (Masked Language Model) involves randomly altering some of the input tokens during training, which the model then learns to predict. During this learning process, the final hidden vector corresponding to the [MASK] token is fed into the final Softmax layer that maps to the model's vocabulary to produce the output. This essentially follows the same approach used for training traditional language models.

In BERT, 15% of the tokens in the entire input are selected to be masked. Of these, 80% are converted to the original MASK tokens, 10% are changed to random tokens that are not the original word, and the remaining 10% are left unchanged. The model then uses the final hidden state vector for each word to predict the original tokens. This only accounts for 1.5% of the entire dataset, so it is believed that this doesn't significantly impact the model's overall understanding of the language. Also, adopting this approach means that the Transformer encoder is forced to maintain contextual information for the entire input sequence, as it doesn't know which words it will need to predict(1).

3.1.2 Next Sentence Prediction (NSP)

Next, let's look into Next Sentence Prediction (NSP). Among various natural language problems, there are tasks like Question Answering (QA) and Natural Language Inference (NLI) that require understanding the relationship between sentences, something that general language models may struggle with. To train

on such relational aspects between sentences, an additional binary classification task called Next Sentence Prediction was introduced. Sentences A and B are selected from the pre-training data, with 50% actually being consecutive sentences (IsNext) and the other 50% being completely unrelated sentences (NotNext). The task is straightforward: simply predict whether the two sentences are consecutive or not. Despite its simplicity, this approach proves to be very effective for tasks like QA and NLI.

3.2 Three Types of Embeddings

To achieve high performance in these two tasks, namely MLM (Masked Language Modeling) and NSP (Next Sentence Prediction), BERT utilizes three types of embeddings. These embeddings assist the model in understanding how words within a sentence or between sentences relate to each other, as well as the significance of the position of each word in the sequence. At this point, it is essential to include 7 key special characters such as [MASK], [SEP], and [CLS] in the vocabulary used for BERT.

3.2.1 Token Embeddings

These are the embeddings for the individual words in the sentence. They serve as the starting point for representing words and are usually learned from a large corpus during the pre-training phase. In the experiment, a tokenizer was created by generating a SentencePiece model with a vocabulary size of 8,000, using the Korean Namuwiki corpus.

3.2.2 Segment Embeddings

Since BERT is designed to handle multiple sentences (Sentence A and Sentence B), segment embeddings are used to distinguish different sentences. They are added to the token embeddings to indicate whether a token belongs to the first sentence or the second. In the experiment, trimming was applied to ensure that both sentences maintained a maximum length, and true/false cases were generated with a 50% probability. If the length of Sentence A exceeded the maximum length, tokens were removed from the beginning, and if the length of Sentence B exceeded the maximum length, tokens were removed from the end.

3.2.3 Positional Embeddings

BERT utilizes the Transformer architecture, which doesn't have a built-in sense of order or sequence. Positional embeddings are added to give the model information about the position of each word within the sequence.

The final input representation for each token is the sum of its token embedding, segment embedding, and positional embedding. This combined embedding is used as the input to the Transformer model.

3.3 Fine-tuning

During the Fine-tuning phase, the self-attention mechanism in the Transformer architecture makes it easier to apply the BERT model to a wide range of natural language problems. BERT's self-attention simultaneously encodes pairs of input strings and applies bidirectional attention, allowing problems that take both pairs of strings and single strings as input to be handled by the same model using the same fine-tuning approach.

Additionally, the introduction of the CLS token at the beginning of the sentence enables the model to solve classification problems like entailment and sentiment analysis. The CLS token is not part of the sentence tokens and is positioned at the very beginning, allowing it to oversee the entire input sequence.

Therefore, during the Fine-tuning phase, all that needs to be done is to apply problem-specific input and output to BERT, and then train all the pre-trained parameters end-to-end for each natural language task.

4 Result

The entire corpus provided consisted of approximately 4 million lines. After converting it into a dataset suitable for BERT pretraining, it was reduced to about 860,000 lines. This dataset, although large at 1.4GB, is still around 100 times smaller than the data typically used for training BERT models. During the experiments, we utilized np.memmap to minimize memory usage and trained the model on only 1/7th of the data. Despite these efforts, the process was time-consuming, and the achieved accuracy was in the range of 0.6, indicating suboptimal performance. While our attempt at creating a very small BERT pre-trained model did not yield significant performance, we observed that numerous optimized versions of BERT continue to emerge. This is not only due to the inherent advantages of BERT but also because even better-performing models like GPT-3 and T5-11B require tremendous computational resources, making them difficult to fine-tune unless backed by large enterprises with extensive capital and equipment.

Currently, active research is being conducted to reduce the size and computational complexity of BERT while maintaining or even improving performance. Prominent models like ALBERT and DistilBERT have emerged as a result. ALBERT uses factorization between the embedding layer and the hidden layer to reduce the model's size and employs cross-layer parameter sharing to reduce the number of parameters while increasing the model's depth. DistilBERT works by compressing the learned layers of BERT, resulting in a model roughly half the size of the original but with similar or slightly reduced performance.

The two-stage approach of BERT involving pretraining and fine-tuning has enabled superior performance even on small datasets, thereby promoting the broad application of transfer learning in NLP. This has also made it easier to fine-tune a general model for various NLP tasks. However, difficulties often arise when fine-tuning BERT-Large models. The model not only learns inaccurate patterns and biases from the training data but also shows unstable training. When fine-tuning the same model on the same dataset with different random seeds, the model's performance shows significant variance. Two hypotheses have been proposed to explain this: Catastrophic forgetting and Small training data size. However, some papers argue that the real causes are the vanishing gradient problem in the early stages of training and generalization issues later on(**?**).

Given these findings, future research should focus on reducing instability during the fine-tuning process.

References

[1] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018.

[2] Marius Mosbach, Maksym Andriushchenko, and Dietrich Klakow. On the stability of fine-tuning BERT: misconceptions, explanations, and strong baselines. *CoRR*, abs/2006.04884, 2020.