Multi-passage BERT: A Globally Normalized BERT Model for Open-domain Question Answering

Zhiguo Wang, Patrick Ng, Xiaofei Ma, Ramesh Nallapati, Bing Xiang
AWS AI Labs
{zhiguow, patricng, xiaofeim, rnallapa, bxiang}@amazon.com

Abstract

BERT model has been successfully applied to open-domain QA tasks. However, previous work trains BERT by viewing passages corresponding to the same question as independent training instances, which may cause in-comparable scores for answers from different passages. To tackle this issue, we propose a multi-passage BERT model to globally normalize answer scores across all passages of the same question, and this change enables our QA model find better answers by utilizing more passages. In addition, we find that splitting articles into passages with the length of 100 words by sliding window improves performance by 4%. By leveraging a passage ranker to select high-quality passages, multi-passage BERT gains additional 2%. Experiments on four standard benchmarks showed that our multi-passage BERT outperforms all state-of-the-art models on all benchmarks. In particular, on the OpenSQuAD dataset, our model gains 21.4% EM and 21.5% $F_1$ over all non-BERT models, and 5.8% EM and 6.5% $F_1$ over BERT-based models.

1 Introduction

BERT model (Devlin et al., 2018) has achieved significant improvements on a variety of NLP tasks. For question answering (QA), it has dominated the leaderboards of several machine reading comprehension (RC) datasets. However, the RC task is only a simplified version of the QA task, where a model only needs to find an answer from a given passage/paragraph. Whereas, in reality, an open-domain QA system is required to pinpoint answers from a massive article collection, such as Wikipedia or the entire web.

Recent studies directly applied the BERT-RC model to open-domain QA (Yang et al., 2019; Nogueira et al., 2018; Alberti et al., 2019). They firstly leverage a passage retriever to retrieve multiple passages for each question. During training, passages corresponding to the same question are taken as independent training instances. During inference, the BERT-RC model is applied to each passage individually to predict an answer span, and then the highest scoring span is selected as the final answer. Although this method achieves significant improvements on several datasets, there are still several unaddressed issues. First, viewing passages of the same question as independent training instances may result in in-comparable answer scores across passages. Thus, globally normalizing scores over all passages of the same question (Clark and Gardner, 2018) may be helpful. Second, previous work defines passages as articles, paragraphs, or sentences. However, the question of proper granularity of passages is still underexplored. Third, passage ranker for selecting high-quality passages has been shown to be very useful in previous open-domain QA systems (Wang et al., 2018a; Lin et al., 2018; Pang et al., 2019). However, we do not know whether it is still required for BERT. Fourth, most effective QA and RC models highly rely on explicit inter-sentence matching between questions and passages (Wang and Jiang, 2017; Wang et al., 2016; Seo et al., 2017; Wang et al., 2017), whereas BERT only applies self-attention layers over the concatenation of a question-passage pair. It is unclear whether the inter-sentence matching still matters for BERT.

To answer these questions, we conduct a series of empirical studies on the OpenSQuAD dataset (Rajpurkar et al., 2016; Wang et al., 2018a). Experimental results show that: (1) global normalization makes QA model more stable while pinpointing answers from large number of passages; (2) splitting articles into passages with the length of 100 words by sliding window brings 4% improvements; (3) leveraging a BERT-based passage
ranker gives us extra 2\% improvements; and (4) explicit inter-sentence matching is not helpful for BERT. We also compared our model with state-of-the-art models on four standard benchmarks, and our model outperforms all state-of-the-art models on all benchmarks.

2 Model

Open-domain QA systems aim to find an answer for a given question from a massive article collection. Usually, a retriever is leveraged to retrieve \( m \) passages \( P = \{P_1, \ldots, P_i, \ldots, P_m\} \) for a given question \( Q = (q^1, \ldots, q^{|Q|}) \), where \( P_i = (p^1_i, \ldots, p^{|p_i|}_i) \) is the \( i \)-th passage, and \( q^k \in Q \) and \( p^j_i \in P_i \) are corresponding words. A QA model will compute a score \( Pr(a|Q, P) \) for each possible answer span \( a \). We further decompose the answer span prediction into predicting the start and end positions of the answer span \( Pr(a|Q, P) = Pr(a_s|Q, P)Pr(a_e|Q, P) \), where \( Pr(a_s|Q, P) \) and \( Pr(a_e|Q, P) \) are the probabilities of \( a_s \) and \( a_e \) to be the start and end positions.

BERT-RC model assumes passages in \( P \) are independent of each other. The model concatenates the question \( Q \) and each passage \( P_i \) into a new sequence \([CLS] p^1_i, \ldots, p^{|p_i|}_i [SEP] q^1, \ldots, q^{|Q|} [SEP]\), and applies BERT to encode this sequence. Then the vector representation of each word position from BERT encoder is fed into two separate dense layers to predict the probabilities \( P_s \) and \( P_e \) (Devlin et al., 2018). During training, the log-likelihood of the correct start and end positions for each passage is optimized independently. For passages without any correct answers, we set the start and end positions to be 0, which is the position for the first token [CLS]. During inference, BERT-RC model is applied to each passage individually to predict an answer, and then the highest scoring span is selected as the final answer. If answers from different passages have the same string, they are merged by summing up their scores.

Multi-passage BERT: BERT-RC model normalizes probability distributions \( P_s \) and \( P_e \) for each passage independently, which may cause incomparable answer scores across passages. To tackle this issue, we leverage the global normalization method (Clark and Gardner, 2018) to normalize answer scores among multiple passages, and dub this model as multi-passage BERT. Concretely, all passages of the same question are processed independently as we do in BERT-RC until the normalization step. Then, \textit{softmax} is applied to normalize all word positions from all passages.

**Passage ranker** reranks all retrieved passages, and selects a list of high-quality passages for the multi-passage BERT model. We implement the passage ranker as another BERT model, which is similar to multi-passage BERT except that at the output layer it only predicts a single score for each passage based on the vector representation of the first token [CLS]. We also apply \textit{softmax} over all passage scores corresponding to the same question, and train to maximize the log-likelihood of passages containing the correct answers. Denote the passage score as \( Pr(P_i|Q, P) \), then the score of an answer span from passage \( P_i \) will be \( Pr(P_i|Q, P)Pr(a_s|Q, P)Pr(a_e|Q, P) \).

3 Experiments

**Datasets:** We experiment on four open-domain QA datasets. (1) OpenSQuAD: question-answer pairs are from SQuAD 1.1 (Rajpurkar et al., 2016), but a QA model will find answers from the entire Wikipedia rather than the given context. Following Chen et al. (2017), we use the 2016-12-21 English Wikipedia dump. 5,000 QA pairs are randomly selected from the original training set as our validation set, and the remaining QA pairs are taken as our new training set. The original development set is used as our test set. (2) TriviaQA: TriviaQA unfiltered version (Joshi et al., 2017) are used. Following Pang et al. (2019), we randomly hold out 5,000 QA pairs from the original training set as our validation set, and take the remaining pairs as our new training set. The original development set is used as our test set. (3) Quasar-T (Dhingra et al., 2017) and (4) SearchQA (Dunn et al., 2017) are leveraged with the official split.

**Basic Settings:** If not specified, the pre-trained BERT-base model with default hyper-parameters is leveraged. ElasticSearch with BM25 algorithm is employed as our retriever for OpenSQuAD. Passages for other datasets are from the corresponding releases. During training, we use top-10 passages for each question plus all passages (within the top-100 list) containing correct answers. During inference, we use top-30 passages for each question. Exact Match (EM) and \( \text{F}_1 \) scores (Rajpurkar et al., 2016) are utilized as the evaluation metrics.
Table 1: Results on the validation set of OpenSQuAD.

<table>
<thead>
<tr>
<th>No.</th>
<th>Model</th>
<th>EM</th>
<th>$F_1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Single-sentence</td>
<td>34.8</td>
<td>44.4</td>
</tr>
<tr>
<td>2</td>
<td>Length-50</td>
<td>35.5</td>
<td>45.2</td>
</tr>
<tr>
<td>3</td>
<td>Length-100</td>
<td>35.7</td>
<td>45.7</td>
</tr>
<tr>
<td>4</td>
<td>Length-200</td>
<td>34.8</td>
<td>44.7</td>
</tr>
<tr>
<td>5</td>
<td>w/o sliding-window (same as (3))</td>
<td>35.7</td>
<td>45.7</td>
</tr>
<tr>
<td>6</td>
<td>w/ sliding-window</td>
<td>40.4</td>
<td>49.8</td>
</tr>
<tr>
<td>7</td>
<td>w/o passage ranker (same as (6))</td>
<td>40.4</td>
<td>49.8</td>
</tr>
<tr>
<td>8</td>
<td>w/ passage ranker</td>
<td>41.3</td>
<td>51.7</td>
</tr>
<tr>
<td>9</td>
<td>w/ passage scores</td>
<td>42.8</td>
<td>53.4</td>
</tr>
<tr>
<td>10</td>
<td>BERT+QANet (fix BERT)</td>
<td>35.5</td>
<td>45.9</td>
</tr>
<tr>
<td>11</td>
<td>BERT+QANet (init. from (11))</td>
<td>36.2</td>
<td>46.4</td>
</tr>
<tr>
<td>12</td>
<td>BERT+QANet (init. from (11))</td>
<td>36.2</td>
<td>46.4</td>
</tr>
</tbody>
</table>

Figure 1: Effect of global normalization.

3.1 Model Analysis

To answer questions from section 1, we conduct a series of experiments on OpenSQuAD dataset, and report the validation set results in Table 1. Multi-passage BERT model is used for experiments.

Effect of passage granularity: Previous work usually defines passages as articles (Chen et al., 2017), paragraphs (Yang et al., 2019), or sentences (Wang et al., 2018a; Lin et al., 2018). We explore the effect of passage granularity regarding to the passage length, i.e., the number of words in each passage. Each article is split into non-overlapping passages based on a fixed length. We vary passage length among {50, 100, 200}, and list the results as models (2) (3) (4) in Table 1, respectively. Comparing to single-sentence passages (model (1)), leveraging fixed-length passages works better, and passages with 100 words works the best. Hereafter, we set passage length as 100 words.

Effect of sliding window: Splitting articles into non-overlapping passages may force some nearby answer spans to lose useful contexts. To deal with this issue, we split articles into overlapping passages by sliding window. We set the window size as 100 words, and the stride as 50 words (half the window size). Result from the sliding window model is shown as model (6) in Table 1. We can see that this method brings us 4.7% EM and 4.1% $F_1$ improvements. Hereafter, we use sliding window method.

Effect of passage ranker: We plug the passage ranker into the QA pipeline. First, the retriever returns top-100 passages for each question. Then, the passage ranker is employed to rerank these 100 passages. Finally, multi-passage BERT takes top-30 reranked passages as input to pinpoint the final answer. We design two models to check the effect of the passage ranker. The first model utilizes the reranked passages but without using passage scores, whereas the second model makes use of both the reranked passages and their scores. Results are given in Table 1 as models (8) and (9) respectively. We can find that only using reranked passages gives us 0.9% EM and 1.0% $F_1$ improvements, and leveraging passage scores gives us 1.5% EM and 1.7% $F_1$ improvements. Therefore, passage ranker is useful for multi-passage BERT model.

Effect of global normalization: We train BERT-RC and multi-passage BERT models using the reranked passages, then evaluate them by taking as input various number of passages. These models are evaluated on two setups: with and without using passage scores. $F_1$ scores for BERT-RC based on different number of passages are shown as the dotted and solid green curves in Figure 1. $F_1$ scores for our multi-passage BERT model with similar settings are shown as the dotted and solid blue curves. We can see that all models start from the same $F_1$, because multi-passage BERT is equivalent to BERT-RC when using only one passage. While increasing the number of passages, BERT-RC without using passage scores decreases the performance significantly, which verifies that the answer scores from BERT-RC are incomparable across passages. This issue is alleviated to some extent by leveraging passage scores. On the other hand, performance from multi-passage BERT without using passage scores increases at the beginning, and then flattens out after passage number is over 10. By utilizing passage scores, multi-passage BERT gets better performance while using more passages. This phenomenon shows the effectiveness of global normalization, which enables the model find better answers by utilizing more passages.

Does explicit inter-sentence matching matter? Almost all previous state-of-the-art QA and RC models find answers by matching pas-
Datasets | Quasar-T | SearchQA | TriviaQA | OpenSQuAD
--- | --- | --- | --- | ---
Models | EM | F₁ | EM | F₁ | EM | F₁ | EM | F₁
--- | --- | --- | --- | --- | --- | --- | --- | ---
DrQA (Chen et al., 2017) | 37.7 | 44.5 | 41.9 | 48.7 | 32.3 | 38.3 | 29.8 | -
R² (Wang et al., 2018a) | 35.3 | 41.7 | 49.0 | 55.3 | 47.3 | 53.7 | 39.1 | 37.5
OpenQA (Lin et al., 2018) | 42.2 | 49.3 | 58.8 | 64.5 | 48.7 | 56.3 | 28.7 | 36.6
TraCRNet (Dehghani et al., 2019) | 43.2 | 54.0 | 52.9 | 65.1 | - | - | - | -
HAS-QA (Pang et al., 2019) | 43.2 | 48.9 | 62.7 | 68.7 | 63.6 | 68.9 | - | -
BERT (Large) (Nogueira et al., 2018) | - | - | - | 69.1 | - | - | - | -
BERT serini (Yang et al., 2019) | - | - | - | - | - | - | 38.6 | 46.1
BERT-RC (Ours) | 49.7 | 56.8 | 63.7 | 70.6 | 62.0 | 67.5 | 51.2 | 59.0
Multi-Passage BERT (Base) | 51.3 | 59.0 | 65.2 | 70.6 | - | - | - | -
Multi-Passage BERT (Large) | 51.1 | 59.1 | 65.1 | 70.7 | 63.7 | 69.2 | 53.0 | 60.9

Table 2: Comparison with state-of-the-art models, where the first group are models without using BERT, the second group are BERT-based models, and the last group are our multi-passage BERT models.

3.2 Comparison with State-of-the-art Models

We evaluate BERT-RC and Multi-passage BERT on four standard benchmarks, where passage scores are leveraged for both models. We build another multi-passage BERT for each dataset by initializing it with the pre-trained BERT-Large model. Experimental results from our models as well as other state-of-the-art models are shown in Table 2, where the first group are open-domain QA models without using the BERT model, the second group are BERT-based models, and the last group are our multi-passage BERT models.

From Table 2, we can see that our multi-passage BERT model outperforms all state-of-the-art models across all benchmarks, and it works consistently better than our BERT-RC model which has the same settings except the global normalization. In particular, on the OpenSQuAD dataset, our model improves by 21.4% EM and 21.5% F₁ over all non-BERT models, and 5.8% EM and 6.5% F₁ over BERT-based models. Leveraging BERT-Large model makes multi-passage BERT even better on TriviaQA and OpenSQuAD datasets.

4 Conclusion

We propose a multi-passage BERT model for open-domain QA to globally normalize answer scores across multiple passages corresponding to the same question. We find two effective techniques to improve the performance of multi-passage BERT: (1) splitting articles into passages with the length of 100 words by sliding window; and (2) leveraging a passage ranker to select high-quality passages. With all these techniques, our multi-passage BERT model outperforms all state-of-the-art models on four standard benchmarks.

In future, we plan to consider inter-correlation
among passages for open-domain question answering (Wang et al., 2018b; Song et al., 2018).

References


