UoR at SemEval-2021 Task 4: Using Pre-trained BERT Token Embeddings for Question Answering of Abstract Meaning

Thanet Markchom, Huizhi Liang
University of Reading
White Knights, Berkshire, RG6 6AH
United Kingdom
t.markchom@pgr.reading.ac.uk, huizhi.liang@reading.ac.uk

Abstract

Most question answering tasks usually focus on predicting concrete answers, e.g., named entities. These tasks can be normally achieved by understanding the contexts without additional information required. In Reading Comprehension of Abstract Meaning (ReCAM) task, the abstract answers are introduced. To understand abstract meanings in the context, additional knowledge is essential. In this paper, we propose an approach that leverages the pre-trained BERT Token embeddings as a prior knowledge resource. According to the results, our approach using the pre-trained BERT outperformed the baselines. It shows that the pre-trained BERT token embeddings can be used as additional knowledge for understanding abstract meanings in question answering.

1 Introduction

Question answering (QA) is one of the machine reading comprehension tasks. The goal is to find an answer of a given question based on a given context. In most QA tasks (Chen et al., 2016; Lai et al., 2017), the answers are commonly concrete words appearing in the contexts. Abstract words, on the other hand, have usually been ignored in such tasks. Unlike concrete words, these abstract words are difficult to understand since they cannot be perceived directly with human senses. To study machine comprehension of abstract meaning, a task called Reading Comprehension of Abstract Meaning (ReCAM) (Zheng et al., 2021) was proposed. Unlike other QA tasks, the answers in ReCAM are abstract words that used to summarise the information in the contexts.

ReCAM is divided into three subtasks. For subtask 1, the abstract answers are in the form of imperceptible words such as 'objective', 'chance', and 'prospective'. Subtask 2 is about nonspecificity. The answers in this subtask are abstract words that represent nonspecific or general concepts such as 'vertebrate'. Subtask 3 focuses on generality of the models developed in the first two subtasks. In this subtask, the model in subtask 1 must be evaluated on subtask 2 data and vice versa. Based on these abstract answers, a machine has to understand the abstract meaning of certain words. This requires an additional knowledge to fulfill the lack of abstract concept information (Bi et al., 2019). For example, given a context which is a passage and a question shown in Table 1. In the example of Table 1, the context contains a passage and a question. The answer 'neglected' does not appear anywhere in the given passage. However, the sentence, 'it gradually fell into disrepair in the late 1980s', provides a clue to answer the question. If the model has a prior knowledge that the word 'disrepair' have a connection with 'neglect', then the correct answer will be chosen.

In this work, we propose to use a pre-trained word/token embedding model to provide external knowledge for abstract meaning understanding. The pre-trained BERT token embedding is chosen in this work due to its good performance in various natural language processing tasks. In our approach, the token embeddings are firstly extracted by the pre-trained BERT model. Presumably, these embeddings are enriched with additional information for understanding the abstract meanings. These embeddings are used as inputs in our approach to predict answers. In this way, the prior knowledge from the pre-trained model can be leveraged in the task of abstract answer prediction.

2 Related Work

A QA task usually requires a machine to understand a context to answer a question. Typically, a context is a passage and an answer usually appear somewhere in a given passage. Several ap-

Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021), pages 799–804
Bangkok, Thailand (online), August 5–6, 2021.

proaches have been proposed to compute a combined passage-question representation and then use it to find an answer. The DeepLSTM Reader (Hermann et al., 2015) uses a deep long-term short-term memory (LSTM) encoder to find a representation of a concatenated passage-question text. The Attentive Reader (Hermann et al., 2015) attentively aggregates words to compute a passage representation based on a question. The Attentive Reader was modified in (Chen et al., 2016) where a deep LSTM module with the dot product attention function were replaced by a shallow bi-directional LSTM module with the bi-linear attention function. Recently, Dhingra et al. (2017) proposed a model called Gated-Attention (GA) Reader. It combines a multi-hop architecture and a novel attention mechanism to learn the representations. These models have shown good performances in question answering when answers are concrete words. However, in ReCAM task, the answers are abstract. Only the given contexts may not suffice to answer the questions.

Recently, word/token embeddings from pre-trained models have been popularly used in many applications. They are capable of providing additional knowledge from the resources they were pre-trained. Several pre-trained models have been proposed in the past few years such as GloVe (Pennington et al., 2014) and ELMo (Peters et al., 2018). Many models have also been developed from the transformer architecture (Vaswani et al., 2017), e.g., BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019) and DistilBERT (Sanh et al., 2020). They have been highly successful in pre-training token embeddings for various downstream tasks including QA. However, there has been no work where such models are applied in a QA task that focuses on imperceptibility and nonspecificity abstractness.

3 Using pre-trained token embeddings for a QA task

Each sample in the dataset of both subtasks consists of three components: passage, question and options. Each passage is a long text that is used to provide context for answering a question. Each question is a passage-summarized text containing one special token, *@placeholder*. This token is a representative of a word (an answer) that should be filled to complete the text. The correct answer must be selected from five possible options provided in each sample. Table 1 shows one example in subtask 1. In the table, the first row is the passage providing a context for this sample. The second row is the question, which is a sentence that summarises the given passage. It contains one placeholder indicated by @placeholder token. This placeholder should be replaced by an abstract word corresponding to the passage. The third row is the list of options that can be selected to replace the placeholder in the question. The correct options is marked in bold which is 'neglected' in this case. The last row is the label indicates the answer of this example ranging from 0 (the first option) to 4 (the last option).

Given a context (a passage and a question containing a placeholder token), a placeholder embedding extracted from the pre-trained BERT model should be able to guide an answer. Due to the fact that the BERT model considers token contexts to learn token embeddings, a placeholder embedding should comprise information of its context. That means any word or token with a similar embedding should also be in the same context as a placeholder as well. Thus, to find the correct answer from a list of options, every option embedding extracted from the pre-trained BERT model is compared with a placeholder embedding. Then, any option with the most similar embedding compared to a placeholder embedding should be an answer. Accordingly, we propose an approach that considers similarity between placeholder and option embeddings to predict answers. Several metrics such as dot product, cosine similarity and euclidean distance can be considered to measure the similarity. The selected metric in this work is described in Section 4.

To extract placeholder and option embeddings in each sample, a question and a passage are firstly concatenated in the following form:

[CLS] + Question + [SEP] + Passage + [SEP]

This form is conventional in BERT framework. Normally, a '[CLS]' token is added for a classification purpose. It is usually represented a sentence level embedding. However, it is not utilized in our approach since we utilized embeddings in a token level. Two '[SEP]' tokens in the form are used to separate a question and a passage. Similarly, for every option in each sample, both '[CLS]' and '[SEP]' tokens are added as follows:

[CLS] + Option + [SEP]

Then, the pre-trained BERT model is used to extract embeddings from these prepared inputs.

Passage	The Trainspotting author has agreed to become patron of The Leith Theatre and launch a new fundraising drive. The Leith Theatre Trust took over the lease of the art deco venue from City of Edinburgh Council last year ... However, it gradually fell into disrepair in the late 1980s and eventually had to be closed down by the council ...
Question	Irvine Welsh is to spearhead a campaign aimed at reviving a @*placeholder* theatre in Leith 30 years after its last show.
Options	**(A) neglected**, (B) renewed, (C) lavish, (D) revised, (E) proposed
Label	0

Table 1: An example of a passage, a question, and five options. The task is to select the correct answer (bold) for replacing @placeholder

The placeholder embedding is extracted from the question-passage concatenated input and each option embedding is extracted from each option input. These embeddings are used as inputs of the proposed approach. As for targets, all labels given are converted to one-hot vectors. With the placeholder and options embeddings as inputs and the one-hot vectors as targets, our proposed approach learn how to predict the probabilities that each option is an answer for each sample. Figure 1 shows the input and target generated from an example shown in Table 1. The pre-trained BERT model is used to extract embeddings of the placeholder and option tokens in the prepared format. The embeddings are the hidden weights of the given tokens from the last hidden layer of the pre-trained BERT model. The label 0 is converted to a one-hot vector $[1, 0, 0, 0, 0]$. These embeddings are then passed through the prediction model. This prediction model consists of six learning modules for learning placeholder and five option embeddings in each sample. These modules consist of a dense layer with an output size 768 and a tanh activation function. They take a 768-dimensional pre-trained BERT token embedding and outputs the fine-tuned embedding with the same size. For each placeholder embedding, the fine-tuned embedding is produced by the placeholder embedding module. Similarly, for each option embedding, the fine-tuned option embedding is also obtained from the corresponding option learning module. However, using separate option learning module for each option may cause a bias in selecting some options at the end. We therefore propose to use shared-weight modules for learning all fine-tuned option embeddings simultaneously. In other words, all options in each sample are learned by the modules that share the same weights. After that, for each option, the similarity between the fined-tuned placeholder and the fine-tuned option embedding is computed. All of the similarities from all options are then concatenated to form a vector with the size 5. A softmax activation function is applied on the concatenated vector to produce the final output with the same size. This output vector represent the probabilities that each option is an answer of a given sample. The entire proposed approach is illustrated in Figure 2. In the figure, p denotes the fine-tuned placeholder embedding while o_0, o_1 and o_4 denote the first, the second and the last option embedding respectively. s denotes a similarity function. $s(p, o_i)$ is the similarity between p and any o_i.

4 Experimental setup

The pre-trained BERT model used in this work is BERT-Base-Uncased [1] (Turc et al., 2019). All inputs were converted to lower case and tokenized by BERT-Uncased tokenizer. The prediction model was trained using RMSprop optimizer for 100 epochs with the learning rate set to 0.001. A categorical cross entropy was used as a loss function. To avoid over-fitting, both L1 and L2 regularizers were added at the dense layer in both placeholder and option learning modules. The regularization factors were set to 0.001. To select the similarity metric, we examined three functions, i.e., dot product (s_d), cosine similarity (s_c) and euclidean-distance-based similarity (s_e) computed by

$$s_d(p, o) = p \cdot o \tag{1}$$

$$s_c(p, o) = \frac{p \cdot o}{||p||||o||} \tag{2}$$

$$s_e(p, o) = \frac{1}{1 + ||p - o||} \tag{3}$$

[1] https://github.com/google-research/bert

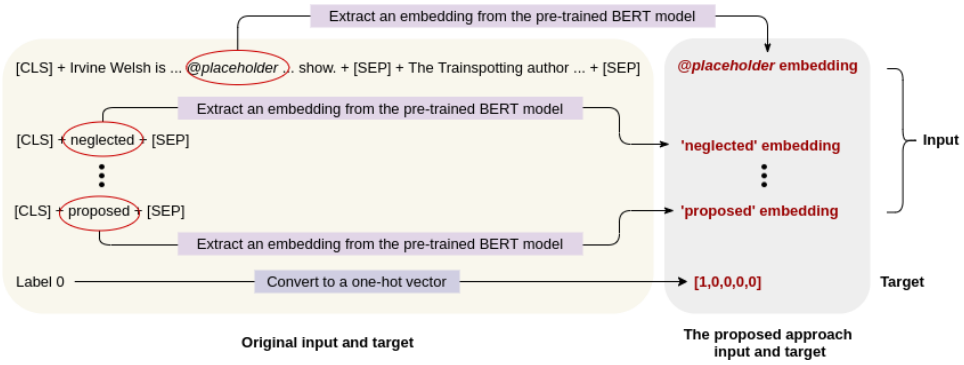

Figure 1: The input and target generated from an example shown in Table 1

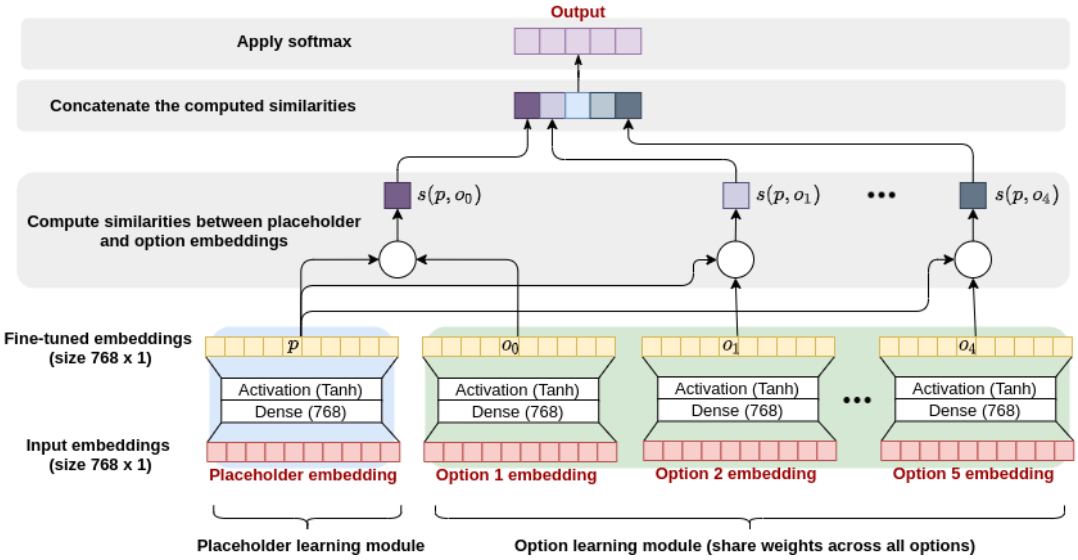

Figure 2: The proposed approach framework

where p denotes a placeholder embedding and o denotes an option embedding. We applied these metrics in the proposed approach and performed experiments on the development sets provided. The results are shown in Table 2. As in the table, the proposed approach using a dot product performed best compared to the others. Hence, a dot product was used in the proposed approach when it was applied on the trial and test sets

Similarity metric	Accuracy	
	Subtask 1	Subtask 2
s_d	0.48	0.46
s_c	0.34	0.31
s_e	0.33	0.31

Table 2: Evaluation results of the proposed approach using different similarity metrics on the development sets

4.1 baselines

We compared the proposed approach named as **BERT-S** with other three baselines as follows:

- **MLP**: a multilayer perceptron (MLP) binary classifier. It consists of two hidden layers with the size 512 and 128 with a rectified linear unit (ReLU) activation function. The output is produced by an output layer with the size 1 and a sigmoid activation function. Each input is a concatenated placeholder-option embedding and its label is 1 if the option is an answer and 0 otherwise. For every sample, an answer is chosen from the concatenated placeholder-option embedding that gives the highest prediction value. A binary cross entropy was used as a loss function. The model was trained by RMSprop optimizer for 100 epochs with the learning rate 0.01.
- **GA**: the GA Reader proposed in (Dhingra et al., 2017). It uses a novel multiplicative

Dataset	Method	Accuracy			
		Subtask 1	Subtask 2	Subtask 3-1	Subtask 3-2
Trial	MLP	0.44	0.28	0.20	0.19
	GA	0.32	0.25	0.24	0.25
	BERT-C	0.26	0.14	0.21	0.21
	BERT-S	**0.50**	**0.30**	**0.29**	**0.27**
Test	MLP	0.47	0.46	0.27	0.25
	GA	0.19	0.21	0.20	0.19
	BERT-C	0.38	0.34	0.24	0.25
	BERT-S	**0.50**	**0.49**	**0.34**	**0.39**

Table 3: Evaluation results on the trial and test sets of all subtasks

gating mechanism, combined with a multi-hop architecture to learn token embeddings based on the given question. The GA Reader obtains state-of-the-art results on three QA benchmarks. However, those benchmarks only focuses on concrete concept answering.

- **BERT-C**: a modified version of the proposed approach. Instead of using similarities between the fine-tuned placeholder and option embeddings, scalar outputs from an MLP are used. After the fine-tuned placeholder and option embeddings are obtained, each fine-tuned option embedding is concatenated to the fine-tuned placeholder embedding. Then, the concatenated embedding is fed to a MLP module. This MLP module consists of three dense layers with the output size 512, 128 and 1 respectively. The scalar outputs from all options are then concatenated and proceeded as in **BERT-S**. The other settings were also set as same as the settings in **BERT-S**.

5 Results

The proposed approach was evaluated on the trial and test sets of all three subtasks. The evaluation metric is accuracy. The results of subtask 1 and 2 are shown in Table 3. There are two results of subtask 3 shown as subtask 3-1 and subtask 3-2. Subtask 3-1 is the proposed approach trained on subtask 1 but evaluated on subtask 2. Similarly, subtask 3-2 is the proposed approach trained on subtask 2 but evaluated on subtask 1. As shown in the table, the proposed approach performed better in subtask 1 compared to subtask 2. It means that the pre-trained BERT model applied in this work is capable of understanding imperceptible words. In contrast, it is not suitable for nonspecific abstract words as it performed poorer in subtask 2. For subtask 3, the proposed approach is not generalized across imperceptible and nonspecific concepts. This can be implied by the significant drop in accuracy in both subtask 3-1 and 3-2. However, compared with the baselines, the proposed approach performed better in all subtasks.

6 Conclusion

In this work, we propose to use the pre-trained BERT token embeddings for QA of abstract meaning. These embeddings are additional information that help understanding abstract meanings in the tasks. According to the results, our approach outperformed the baselines in every subtask. It means that the pre-trained BERT model is effective in QA of abstract meaning that focus on imperceptibility and nonspecificity. For the future work, it is worth to fine-tune the embeddings from the pre-trained model in an end-to-end manner. Other resources, e.g., semantic graphs, are also worth considering to provide more information for machine comprehension of abstract meanings.

References

Bin Bi, Chen Wu, Ming Yan, Wei Wang, Jiangnan Xia, and Chenliang Li. 2019. Incorporating external knowledge into machine reading for generative question answering.

Danqi Chen, Jason Bolton, and Christopher D. Manning. 2016. A thorough examination of the CNN/Daily Mail reading comprehension task. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2358–2367, Berlin, Germany. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of

deep bidirectional transformers for language understanding.

Bhuwan Dhingra, Hanxiao Liu, Zhilin Yang, William W. Cohen, and Ruslan Salakhutdinov. 2017. Gated-attention readers for text comprehension.

Karl Moritz Hermann, Tomáš Kočiský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. 2017. RACE: Large-scale ReAding Comprehension dataset from Examinations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 785–794, Copenhagen, Denmark. Association for Computational Linguistics.

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.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2020. DistilBERT, a distilled version of bert: smaller, faster, cheaper and lighter.

Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Well-read students learn better: On the importance of pre-training compact models. *arXiv preprint arXiv:1908.08962v2*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need.

Boyuan Zheng, Xiaoyu Yang, Yuping Ruan, Quan Liu, Zhen-Hua Ling, Si Wei, and Xiaodan Zhu. 2021. SemEval-2021 task 4: Reading comprehension of abstract meaning. In *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021)*.