Roof-BERT: Divide Understanding Labour and Join in Work

Anonymous ACL submission

Abstract

Recent work on enhancing BERT-based language representation models with knowledge graphs (KGs) and knowledge bases (KBs) has promising results on multiple NLP tasks. State-of-the-art approaches typically integrate the original input sentences with triples in KGs, and feed the combined representation into a BERT model. However, as the sequence length of a BERT model is limited, the framework can not contain too much knowledge besides the original input sentences and is thus forced to discard some knowledge. The problem is especially severe for those downstream tasks that input is a long paragraph or even a document, such as QA or reading comprehension tasks. To address the problem, we propose Roof-BERT, a model with two underlying BERTs and a fusion layer on them. One of the underlying BERTs encodes the knowledge resources and the other one encodes the original input sentences, and the fusion layer like a roof integrates both BERTs' encodings. Experiment results on QA task and GLUE benchmark reveal the effectiveness of the proposed model.

1 Introduction

While BERT model dominates multiple benchmark datasets, studies on incorporating extra knowledge with Language Models (LMs) for advancing language understanding sprung up (Zhang et al., 2019; Liu et al., 2019; Wang et al., 2020). The sources of the extra knowledge are mostly KGs and KBs providing rich knowledge facts and benefiting language understanding. For example, ERNIE (Zhang et al., 2019) employs TransE (Bordes et al., 2013) to encode entity information, and concatenate them with the token embedding to feed into a fusion layer. Despite the success on GLUE benchmark, due to the concatenation on token level, ERNIE is not able to consider textual knowledge representation. On the other hand, K-BERT (Liu et al., 2019) convert knowledge triples into textual forms and inject them into the input sentences, forming a tree representation to feed into a BERT. However, this kind of approaches can only consider pretty limited knowledge besides the input sentences due to intrinsic limitation of input length of BERT (512 tokens).

Accordingly, we propose **Roof-BERT**, a model with two underlying BERTs and a fusion layer, Transformer encoder (Vaswani et al., 2017), as the **Roof** on top of them. Roof-BERT encodes the text input with one of the underlying BERTs and encode the knowledge information with the other BERT, and integrate both embeddings with a fusion layer for further downstream tasks. Through the structure, our model allows more information from both the original text and knowledge information. In addition, if memory permits and the necessity of long input, employing multiple BERTs (more than two) is also accessible through the structure.

Although the proposed idea is intuitive, there are still several critical challenges which need to be addressed:

(1) What is a appropriate model for a Roof? And how does the Roof distinguish individual outputs from two underlying BERTs?

(2) How many layers are enough for the roof to fuse the outputs from two underlying BERTs? There could be a trade-off between computational resources and performance.

(3) Due to different model complexities, necessary converge time for Roof may be different from converge time for BERTs. How to address the issue during the training phase?

(4) Although through our proposed architecture, the space for knowledge can be as long as 512 tokens, it is still limited. Thus precise knowledge selection and effective representation would be crucial for the performance.

We investigate various factors and propose the corresponding solutions to these challenges, described in detail in the following sections.

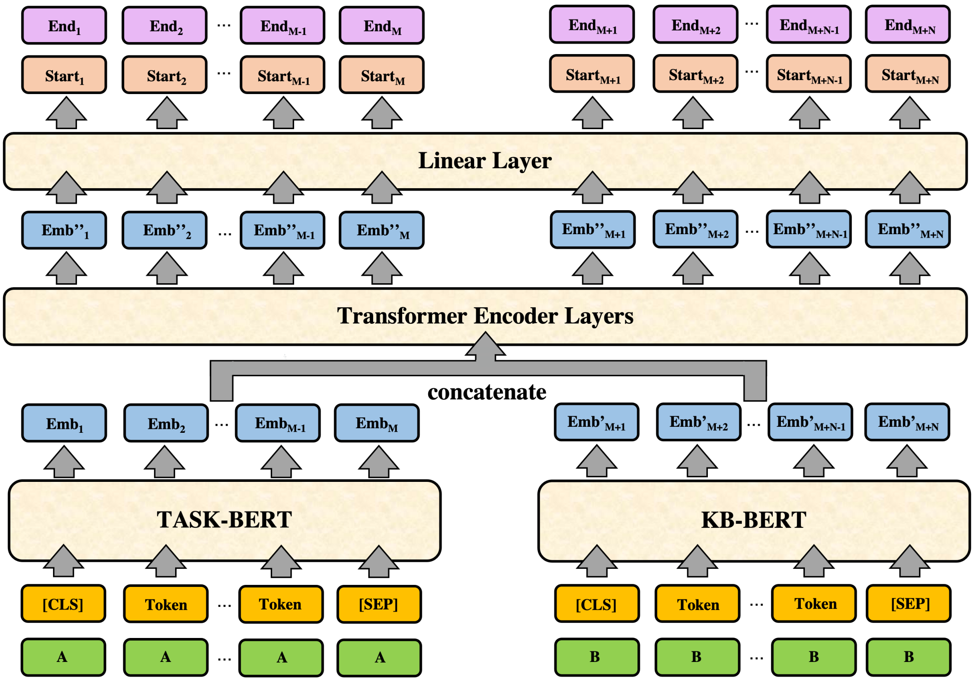

Figure 1: The overall architecture of Roof-BERT for QA task: The input of Roof-BERT contains input tokens and segmentation tokens, which are the yellow and green blocks respectively. The linear layer, in the case of QA task, outputs two digits, which are the probabilities of the start and end positions of the answer, for every position. In the case of NLU tasks in GLUE, we input the average mean of the transformer encoder layers' output into the linear layer, the linear layer outputs the probability distribution of the labels.

We conduct experiments on the QA task (Rajpurkar et al., 2016) and GLUE benchmark. Experiment results reveal that integrating knowledge by Roof-BERT structure significantly outperforms the results of using only one BERT to integrate both original input sentences and knowledge.

Overall, our contributions can be summarized as follows:

- We propose a BERT-based framework, namely Roof-BERT, which encodes knowledge and input sentences with two separate BERTs. Promising results of QA task and GLUE benchmark are provided.

- Roof-BERT address the problem of input length limitation of BERT. We believe it can also contribute to other NLP tasks where much knowledge or long context comprehension is needed.

- We show that precise knowledge selection and effective representation are critical to advance the performance for various downstream NLP tasks.

2 Related Work

Many researches have worked on integrating KB/KG for enhanced language representation.

Before strong pre-trained LMs such as BERT were proposed, several works have studied joint representation learning of words and knowledge. (Wang et al., 2014) combines knowledge embeddings and word vectors. (Toutanova et al., 2015) utilizes the convolutional neural network to capture the compositional structure of textual relations, and jointly optimizes entity, KBs, and textual relation representations. Both of them are also based on the concept of word2vec (Mikolov et al., 2013) and TransE (Bordes et al., 2013).

After Google Inc. launched BERT in 2018, researches on integrating KBs/KGs gradually focus on optimization with pre-trained LMs. ERNIE (Zhang et al., 2019), one of the early studies, encodes knowledge information in KGs by knowledge embedding model TransE (Bordes et al., 2013) trained on Wikidata and refines pre-training of BERT via named entity masking and phrase masking. K-BERT (Liu et al., 2019) proposed injecting knowledge into the text to form a sentence tree without using a pre-training by-self model for knowledge embeddings and adopted soft-position embeddings and visible matrix for structural information and prevention of diverting the sentence from its correct meaning. Based on these works, KEPLER (Wang et al., 2020) jointly optimizes the knowledge embeddings and mask language modeling objectives on pre-training LMs.

There are also some related work relying on two BERTs working together. For example, Sentence-BERT (Reimers and Gurevych, 2019), propose a model to derive sentence embeddings via BERTs and use a classifier to judge the similarity of two sentences. DC-BERT (Zhang et al., 2020), a decoupled contextual encoding framework to address the efficiency of information retrieval, use an online BERT to encode the question only once, and an offline BERT which pre-encodes each document and caches their encodings.

3 Methodology

In this section, we detail the overall framework of Roof-BERT presented in Figure 1 and the input formats for Roof-BERT, which are sentences pairs and selected triples from KB.

3.1 Model Architecture

As shown in Figure 1, the entire model architecture of Roof-BERT contains three stacked modules: (1) the Underlying BERT model, responsible for encoding tokens to meaningful representations, (2) the Fusion layer, responsible for combining information from the Underlying BERT model, and (3) the Prediction layer, responsible for the further downstream task, in our case, the QA task and the common Natural Language Understanding (NLU) tasks.

Underlying BERT model. The Underlying BERT model is composed of two independent BERT models, denoting as TASK-BERT and KB-BERT. TASK-BERT encodes the tokenized passages, which are identical to those input for a single BERT on each downstream task, into embeddings. KB-BERT encodes the tokenized triples from KBs to embeddings. Both embeddings are then concatenated and fed to the Fusion layer as input.

Fusion layer. We choose Transformer Encoder (TE) layers (Vaswani et al., 2017) as our Fusion layer due to its self-attention mechanism. The input of the Fusion layer is the concatenation of output embeddings from TASK-BERT $\mathbf{Emb} \in \mathbb{R}^{M\times d}$ and the output embeddings from KB-BERT $\mathbf{Emb}' \in \mathbb{R}^{N\times d}$, where d is the dimension or the hidden size of word embeddings and M and N are the length of the tokenized passage of question and paragraph and tokenized triples from KBs, respectively.

Prediction layer. The Prediction Layer is simply a Linear NN Layer, which is responsible for transforming high-dimension embeddings into appropriate logits for prediction and inference. The input of the prediction layer is the output embeddings of the Fusion layer $\mathbf{Emb}'' \in \mathbb{R}^{(M+N)\times d}$; while the output of QA task is $\mathbf{logits} \in \mathbb{R}^{(M+N)\times 2}$, and the two dimensions of the output logits in each position are the start and end logits, which stands for the probability of whether the position is the start or the end position of the answer. On the other hand, in other NLU tasks, the output embeddings of Fusion layer are compressed to only one sequence length $\mathbf{AvgEmb} \in \mathbb{R}^{d}$; while the output $\mathbf{logits} \in \mathbb{R}^{e}$, and e is based on the number of class of predicted label.

The model parameters are updated by minimizing the cross-entropy loss between the output logits and the ground truths.

3.2 Input Format for TASK-BERT

In this paper, we test the capability of Roof-BERT on QA downstream task and NLU tasks of GLUE. We follow the format of the input of the major approach for BERT. Each sentence pair from the dataset contains two passages. Both passages will be tokenized and will be concatenated with a [SEP] token as input for TASK-BERT.

Since BERT has a 512 limitation on input length, we set the maximum length for our question passage and paragraph passage. If the length of the passage is shorter than the maximum length, which is mostly the case in the question passage, [PAD] tokens will be added at the end of the concatenated passage to fix the length of every input. If the length of the passage is longer than the maximum length, which is mostly the case in paragraph passages, we will truncate the passage.

3.3 Input Format for KB-BERT

We choose KBs as our external information for our approach. KB contains triples and each triple consists 3 components, which are head, relation, and tail, each corresponds to subject, relation, and object in a sentence respectively.

The triples will be selected by a heuristic algorithm (string match), which will select the triple if its head exists in the paragraph passage in the input of TASK-BERT. The selected triples, will then con-

Sample triples (head—relation—tail)
1. Bill Gates—founder—Microsoft
2. Bill Gates—alumni—Harvard
3. Elon Musk—founder—SpaceX

Expand 0
[SEP] Bill Gates founder Microsoft
[SEP] Bill Gates alumni Harvard
[SEP] Elon Musk founder SpaceX

Expand 1
[SEP] Bill Gates is a founder of Microsoft
[SEP] Bill Gates is a alumni of Harvard
[SEP] Elon Musk is a founder of SpaceX

Expand 2
[SEP] Bill Gates is a founder of Microsoft, he is a alumni of Harvard
[SEP] Elon Musk is a founder of SpaceX

Figure 2: Example of three types of expansion. Note that we use Chinese KB; this example is for demonstration.

catenate one after another, and are separated with [SEP] token.

As shown in Figure 2, we proposed three type of expansions, which are denoted as **Expand 0, 1, 2**, for our selected triples. **Expand 0** simply concatenate the components in the triple as a unit, and it will concatenate after the previous unit. **Expand 1** will further add "的", a Chinese token, between head and relation, and add "是", a Chinese token, between relation and tail to form a natural sentence (Agarwal et al., 2021), which is equivalent to "head is a relation of tail" in English. The sentence, which is also the unit, will then be concatenated after the previous unit. **Expand 2** is a refined version of **Expand 1**; if the current head of the selected triple is identical to the head of the previous unit, the head will be replaced by a pronoun, and merge with the previous unit with a comma, forming a larger sentence/unit.

4 Experiments

In this section, we present the details of fine-training Roof-BERT and the fine-tuning results with different KBs and settings. The estimated number of parameters is around 200 230M depends on the depth of our fusion layer.

4.1 Parameter Settings

For QA task, conformed to the input length limit of BERT, we set the maximum length of question and paragraph as 59 and 450 respectively so that the total length of the input token length would be len([CLS]) + len(Question passge token) + len([SEP]) + len(Paragraph passage token) + len([SEP]) = 1 + 59 + 1 + 450 + 1 = 512.

We find the following setting values work well on the QA datasets, i.e., batch size: 16, learning rate (AdamW): $3e^{-5}$. In addition, we adopt linear learning rate decay scheduler. For the number of epochs, since we are fine-tuning our model on QA downstream task, the number of the epoch is set 1 with the training loss and accuracy converging properly.

For the NLU tasks of GLUE, we set the maximum length of sentence pair as 512, including one [CLS] token and two [SEP] tokens, if the length of sentences doesn't reach 512, we add the [PAD] tokens at the end of the sentences until the length of sentences reaches 512.

We find the following setting values work well on the GLUE benchmark, i.e., batch size: 16, learning rate (AdamW): $2e^{-5}$. We use cosine learning rate decay scheduler. The training epochs are fixed at 5 for fine-tuning our proposed model.

4.2 Dataset

For QA task, we evaluate Roof-BERT and baseline model on DRCD dataset (Shao et al., 2019). For other NLU tasks, we evaluate our model on General Language Understanding Evaluation (GLUE) benchmark (Wang et al., 2018), which is used in ERNIE (Zhang et al., 2019). GLUE is a multi-task benchmark for NLU consisting 11 tasks, and we use the following 8 tasks of GLUE to evaluate Roof-BERT and compare it with baseline models. These tasks adopt different evaluation metrics depending on their purpose.

DRCD. The Delta Reading Comprehension Dataset is an open-source Chinese QA dataset, which is composed of paragraphs from Wikipedia articles and questions generated by annotators. The ground truths of each Question-Paragraph pair are the start and end position of the answer. The result will be evaluated by exact match (EM) score metrics.

GLUE. We select the following 8 tasks of GLUE benchmark: (1) SST-2, a sentiment task using accuracy as metrics, (2) CoLA, a acceptability task using Matthews Corr. as metrics, (3) MRPC, a paraphrase task using F1 score as metrics, (4) STS-B, a sentence similarity task using Pearson-Spearman Corr. as metrics, (5) QNLI, a Natural Language Inference (NLI) task using accuracy score as metrics, (6) QQP, a paraphrase task using F1 score as metrics, (7) RTE, a NLI task using accuracy as metrics, (8) MNLI, a NLI task using accuracy as metrics.

4.3 Knowledge Base

We employ two Chinese KBs, HowNet and CN-DBpedia which are refined and used in K-BERT (Liu et al., 2019) for QA task. Each triple in both KBs holds head, relation, and tail. We also employ KELM Corpus to evaluate our model on common NLU tasks of GLUE benchmark.

CN-DBpedia. The CN-DBpedia (Xu et al., 2017) is a large-scale structured encyclopedia developed and maintained by the Knowledge Workshop Laboratory of Fudan University. It has extended to fields such as law, industry, finance, and medical care, providing supporting knowledge services for intelligent applications in various industries. The refined CN-DBpedia contains around 5M triples.

HowNet. The HowNet (Dong et al., 2006) is a large-scale KB containing Chinese concepts and vocabulary. Each entity is annotated with semantic units called sememes. The refined HowNet contains a total of 52,576 triples.

KELM. The KELM Corpus (Lu et al., 2021) consists of the entire Wikidata KG as natural text sentences, it contains around 15M sentences converted from KG's triples, the sentences are like **Expand 2** shown in Figure 2.

4.4 KB Format

It is intuitive that knowledge representation will effect the performance of models by yielding differentiating quality of language understanding. Besides, which knowledge to select is also an essential issue. Therefore, we test 6 different formats of input of KB-BERT, which are the combinations of 3 kinds of knowledge representation with 2 kinds of knowledge selection.

Representation. We represent the knowledge from KBs with 3 types of Expand mentioned in Section 3.3 and demonstrated in Figure 2, and compare these representations with model performance.

Selection. As mentioned in Section 3.3, triples in KB are selected if its head exists in the paragraph passage no matter its tail exists in the paragraph, and we denote this selection as Tail_nonExist. The other kind of selection is denoted as Tail_Exist, meaning that both the head and the tail of the selected triple exist in the paragraph passage. Checking whether the tail exists in the paragraph passage is a simple way to ensure the selected triples are more likely to relate to the paragraph passage.

The result shows that the combination of **Expand 2** and **Tail_Exist** yields the best performance.

4.5 Other Setting

We also investigate the following factors, which influence the fusion efficiency, through experiments.

Segmentation. Since the input of the fusion layer contains outputs from both underlying BERT models, where they contains their own positional information, the fusion layer can hardly distinguish these two parts. Thus, we add segmentation tokens to the input of KB-BERT and TASK-BERT, and test 2 formats segmentation token to find out the better way to address this problem.

The **first** type of segmentation of KB tokens and padding tokens in KB-BERT are different, using A and B respectively; while the segmentation of question, paragraph, and padding tokens in TASK-BERT are A, B, A respectively. The first type of segmentation aims to separate the content of tokens in a single BERT.

As showed in Figure 1, the **second** type of segmentation of every token in KB-BERT is set to A, that is, the segmentation of KB tokens and padding tokens are all A; while the segmentation of every token in TASK-BERT is set to B, that is, the question, paragraph, and padding tokens are all B. The second type of segmentation aims to separate the content of tokens of two BERT.

The result shows that the **second** type generates a better performance.

Fusion layer. As mentioned in Section 3.1, we use TE layers as our Fusion layer. To answer the questions of 'whether the more layer the better on performance?' and 'whether loading pre-trained weight to Fusion layer helps prediction?', we test different number of TE layers under same setting; we also test TE layers initialized by random weights and pre-trained BERT_{base}, and TE layers are updated during training.

The result shows that 'more layers do not always lead to better performance' and 'adopting pre-trained weight outperforms random weights'.

Learning Rate of TE. Due to different model complexities of BERTs and the fusion layer, we found that necessary converge time for the fusion layer (less complex) would be much more than BERTs (more complex). How to make the two types of models converge at the same time would be a challenge, our proposed solution is to use different learning rates for them: the learning rate of fusion layer is increased to be much larger than the learning rate of BERTs under the same training process. Thus, we investigate different learning rates

Model	KB	TE Initialization	EM score
BERT$_{base}$-chinese	-	-	76.08
Roof-BERT	HowNet	BERT$_{base}$-chinese	77.45
Roof-BERT	CN-DBpedia	BERT$_{base}$-chinese	**77.59**
Roof-BERT	HowNet	random weight	76.31
Roof-BERT	CN-DBpedia	random weight	76.61

Table 1: Results of Roof-BERT and baseline on QA tasks (%) with different KBs and initialization

Model	KB	SST-2	CoLA	MRPC	STS-B	QNLI	QQP	RTE	MNLI-m
BERT$_{base}$	-	93.3	52.1	88.0	**85.0**	90.5	**71.2**	66.4	**84.6**
ERNIE	Wikidata	**93.5**	52.3	88.2	83.2	**91.3**	71.2	68.8	84.0
Roof-BERT	KELM	93.0	**54.4**	**89.0**	84.2	90.6	70.3	**69.0**	84.3

Table 2: Results of Roof-BERT and baselines on eight datasets of GLUE benchmark (%)

on the Fusion layer, e.g. 5 times, 10 times than the Underlying BERTs' learning rate, and make a comparison with using the same learning rate on the entire model.

The result shows that using **10** times learning rate of the Underlying BERT model on the Fusion layer achieves the best performance.

4.6 Baseline

In this paper, we compare Roof-BERT to three baseline: BERT$_{base}$-chinese, BERT$_{base}$-uncased (Devlin et al., 2018) and ERNIE (Zhang et al., 2019). BERT$_{base}$-chinese is pre-trained on WikiZh; BERT$_{base}$-uncased, denoted as BERT$_{base}$, is pre-trained on the BookCorpus and English Wikipedia; ERNIE is pre-trained on English Wikipedia for large-scale textual corpora and Wikidata for KGs. We use BERT$_{base}$-chinese as the baseline for QA task, and use BERT$_{base}$ ERNIE as the baseline tasks in GLUE. Both BERT models are fine-tuned without KBs, and the results of baseline ERINE are from results in (Zhang et al., 2019).

4.7 Results

QA performance. We find the following setting performs the best in Roof-BERT: KB format with **Expand 2** and **Tail_Exist**, segmentation with **second** type, TE initialized with **last 4** layers from pre-trained BERT$_{base}$-chinese with **10 times** of the Underlying BERT model's learning rate.

As the results are shown in Table 1, with external information from KBs, the performance of EM score on QA task increases **>1.5%**, which presents the benefits of utilizing KBs. The quality and the quantity of knowledge from Cn-Dbpedia might be better than that from HowNet so as to result in a slight difference between their EM score in Table 1.

KB format. We compared the results of 6 different KB formats showed in Table 3 with same setting mentioned in Section 4.7-$QAperformance$ except KB format, including knowledge representation and selection. It is clearly that the KB format with **Expand 2** representation and selected by **Tail_Exist** produces the best performance. However, with a closer look in the contributions of knowledge representation and selection, we found that **Expand 2** yields marginally better results among three types of Expand, producing finer language representation; however, whether the tails of the selected triples are in the paragraph passage matters.

The comparison of knowledge selections is shown in Table 4 with same setting mentioned in Section 4.7-$QAperformance$ except knowledge selection. The result indicates that selecting knowledge with tail existing in paragraph passage improves the EM score even with far less knowledge added, inferring that adding arbitrary information might worsen the performance.

TE initialization. As shown in Table 1, using pre-trained weight on TE for initialization performs better on both KBs; while the EM scores with using random weight for initialization do not increase much compare to our baseline. It indicates that pre-trained weight can improve language understanding.

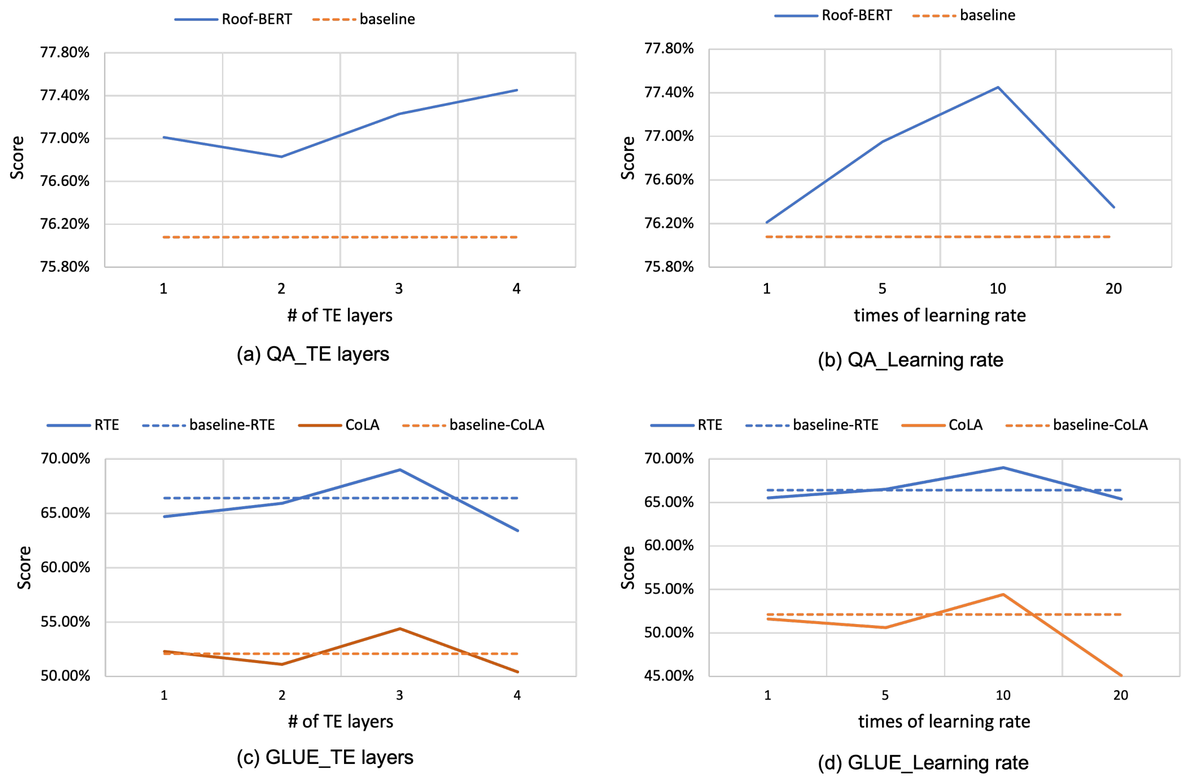

Figure 3: Results of fusion efficiency study on QA and GLUE tasks: The metrics in (a), (b) are both EM scores, while the metrics in (c), (d) are accuracy for RTE and Matthew's Corr for CoLA. In (a) and (c), we set 10 times learning rate of the Underlying BERT model in TE layers; in (b), we set the number of TE layers to 4; in (d), we set the number of TE layers to 3.

Type	Tail_Exist	Tail_nonExist
Expand 0	77.21	77.34
Expand 1	76.92	76.77
Expand 2	**77.45**	77.03

Table 3: Results (EM score %) of different types of KB selections and representations on QA task using HowNet as KB.

KB	Selection	Length	EM Score (%)
HowNet	Tail_Exist	29	**77.45**
HowNet	Tail_nonExist	131	77.03
Cn-Dbpedia	Tail_Exist	24	**77.59**
Cn-Dbpedia	Tail_nonExist	168	76.90

Table 4: Comparison of adopting different knowledge selections, where Length is the average length of the input tokens of KB-BERT (w/o [PAD]) during training.

Number of TE layers. As reported in Figure. 3a and c, the scores reache the maximum when using the last 4 and 3 TE layers from pre-trained BERT as Fusion layer. When using more layers, it consumes much more memory and the scores drop, which might be due to over-fitting; while using less layers, the fusion layer can not integrate language representation with KBs well.

Learning rate of TE. As reported in Figure. 3b, using 10 times the learning rate of the Underlying BERT model's learning rate on 4 TE layers returns the highest EM score increasing >1% compared to using the same learning rate as Underlying BERT model. Similarly, as reported in Figure. 3d, using 10 times the learning rate of the Underlying BERT model's learning rate on 3 TE layers has a 2% performance boost. This shows that adopting a higher learning rate in Fusion layer improves the fusion effectiveness and further enhances prediction. The cause behind the phenomena is that after BERT was pre-trained, a very small learning rate is adequate for fine-tuning on downstream task and requires less epochs for its loss to converge to a minima (global or local); on the contrary, excessive learning rate might not lead our model to convergence. That is to say, it is highly possible that the learning pace of Fusion layer should be faster than

Question	Paragraph	Knowledge	Ans. of Roof-BERT	Ans. of baseline
Which army was the Chinese army that regained lost territory for the first time since the September 18th Incident?	Feng Yuxiang, Ji Hongchang, Fang Zhenwu and others established the Chahar People's Anti-Japanese Allied Army in Zhangyuan on May 26, 1933 and started to attack the Japanese troops in Chahar and Jehol in June, and all of them were expelled from Chahar, which was the first time that the Chinese army had regained lost territory since the September 18th Incident. From May to June in the 34th year of the Republic of China, the Eighth Route Army in the Shanxi-Chahar-Hebei border area launched the Chanan Campaign, captured Huaian, Shuyuan and other county towns, and developed to the Pingsui Road and the Chabei area.	[CLS] The Eighth Route Army is a kind of army [SEP] Allied Army is for Alliance, and is a kind of army [SEP]	The Chahar People's Anti-Japanese Allied Army	The Eighth Route Army
Before Pearl Harbor Attack, which of Japan's most important targets were not in the harbor?	Tokyo received information about spies lurking in Pearl Harbor. There were 9 warships, 3 cruisers, and 17 destroyers parked in the harbor. There were 4 cruisers and 3 destroyers in the dock, while all aircraft carriers were not in the base. The pocket submarine, which was the vanguard of the operation, began to leave the mothership. At 3:42, the U.S. minesweeper USS Condor spotted a periscope in front of Honolulu Harbor, and the destroyer USS Ward fired and dropped depth charges.	[CLS] Aircraft carrier is a kind of weapon [SEP] Periscope is for vision, and is a kind of equipment [SEP] Destroyer is a kind of weapon [SEP] Cruiser is a kind of weapon [SEP]	Aircraft carrier	USS Ward

Figure 4: Case study for QA task (DRCD dataset): This table contains 2 cases, where the ground truths of both cases are matched with those predicted from Roof-BERT and knowledge is from HowNet in these cases. Noted that, content is translated (without meaning loss) since the DRCD dataset and HowNet are in Chinese.

the Underlying BERT model in order not to confine to the individual information each underlying BERT provides but to consider both.

GLUE. We use the following settings to evaluate Roof-BERT on datasets of GLUE: KB representation with **Expand 2**, TE initialized with **last 3** layers from pre-trained BERT$_{base}$ with **10** times of the Underlying BERT model's learning rate.

As the result shown in Table 2, Roof-BERT shows better performance on some datasets, like RTE, CoLA, and MRPC, which means our proposed model can integrate knowledge and contextual embeddings well. Moreover, Roof-BERT achieves comparable or even better results with ERNIE on GLUE benchmark.

The result also shows that Roof-BERT doesn't have a significant effect on other tasks, like SST-2, STS-B, and QNLI. The reason could be the necessity of external information for these tasks, like sentimental analysis, the sentiment of the sentence is determined by the emotional words but not knowledge, this situation also occurs in other researches, like ERNIE and K-BERT. However, our model in these tasks still achieve comparable performance compare to BERT.

5 Case Study

We conduct a case study on the QA task. As result shown in Table 4: In the first case, knowledge provides information that both Chahar People's Anti-Japanese Allied Army and Eighth Route Army are kinds of army, enabling Roof-BERT understand these unseen named entities, which indirectly helps correct prediction. Similarly, in the second case, Roof-BERT also benefits from the added knowledge, making it aware the functions and characters of those named entities (Aircraft carrier, Destroyer...) in the paragraph. On the contrary, without knowledge from KB, our baseline model, BERT$_{base}$-chinese fail to understand the contextual information of given question and paragraph, leading to mis-prediction.

6 Conclusion

In this paper, we propose Roof-BERT to encode knowledge and input sentences with two underlying BERTs respectively and a fusion layer on them as a Roof. Through the architecture, the problem of length limitation of BERT can be eased, so more knowledge and longer input texts can be handled with BERT. Experiment results on QA task and GLUE benchmark are provided to demostrate the model's effectiveness. We also show that precise knowledge selection is also critical under the architecture. Roof-BERT is a very general and powerful method for language understanding, which could easily be applied to other NLP tasks. It would especially benefit tasks which require information from a large range of knowledge or long input texts.

References

Oshin Agarwal, Heming Ge, Siamak Shakeri, and Rami Al-Rfou. 2021. Knowledge graph based synthetic corpus generation for knowledge-enhanced language model pre-training. *arXiv preprint arXiv:2010.12688v2*.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. *Proceedings of NIPS*, pages 2787–2795.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Zhendong Dong, Qiang Dong, and Changling Hao. 2006. Hownet and the computation of meaning. *Citeseer*.

Weijie Liu, Peng Zhou, Zhe Zhao, Zhiruo Wang, Qi Ju, Haotang Deng, and Ping Wang. 2019. K-bert: Enabling language representation with knowledge graph. *arXiv preprint arXiv:1909.07606*.

Yinquan Lu, Haonan Lu, Guirong Fu, and Qun Liu. 2021. Kelm: Knowledge enhanced pre-trained language representations with message passing on hierarchical relational graphs. *arXiv preprint arXiv:2109.04223*.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. *Proceedings of EMNLP*, pages 2383–2392.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084v1*.

Chih Chieh Shao, Trois Liu, Yuting Lai, Yiying Tseng, and Sam Tsai. 2019. Drcd: a chinese machine reading comprehension dataset. *arXiv preprint arXiv:1806.00920v3*.

Kristina Toutanova, Danqi Chen, Patrick Pantel, Hoifung Poon, Pallavi Choudhury, and Michael Gamon. 2015. Representing text for joint embedding of text and knowledge bases. *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1499–1509.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhi. 2017. Attention is all you need. *Proceedings of NIPS*, pages 5998–6008.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

Xiaozhi Wang, Tianyu Gao, Zhaocheng Zhu, Zhengyan Zhang, Zhiyuan Liu, Juanzi Li, and Jian Tang. 2020. Kepler: A unified model for knowledge embedding and pre-trained language representation. *arXiv preprint arXiv:1911.06136v3*.

Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph and text jointly embedding. *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1591–1601.

Bo Xu, Yong Xu, Jiaqing Liang, Chenhao Xie, Bin Liang, Wanyun Cui, and Yanghua Xiao. 2017. Cn-dbpedia: A never-ending chinese knowl- edge extraction system. *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems*, pages 428–438.

Yuyu Zhang, Ping Nie, Xiubo Geng, Arun Ramamurthy, Le Song, and Daxin Jiang. 2020. Dc-bert: Decoupling question and document for efficient contextual encoding. *arXiv preprint arXiv:2002.12591v1*.

Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019. Ernie: Enhanced language representation with informative entities. *Proceedings of ACL*, pages 1441–1451.