Question Answering on SQuAD Dataset

Yuqin Shen

Electrical and Computer Engineering

Duke University, NC 27705

Ys238@duke.edu

Advisor: Patrick Wang

Abstract

This project explores automated question answering model on the SQuAD dataset. It did preliminary data visualization and introduced the google-research BERT model using encoder-decoder architecture and whole word masking. Fine-tuned model checkpoints are used to train on SQuAD 1.1 and it achieves a F1 score of 87.98% on the SQuAD test data.

1. Introduction

Automated Question Answering (AQA) and machine comprehension (MC) have gathered a powerful momentum recently with advances in Deep Learning, which became an essential tool for NLP (Natural Language Processing) and NLU (Natural Language Understanding).

A question answering (QA) system is a system designed to answer questions posed in natural language. Some QA systems draw information from a source such as text or an image in order to answer a specific question. These "sourced" system can be partitioned into two major subcategories:

(1) Open domain:

The questions can be virtually anything but aren't focused on specific material.

(2) Closed domain:

The questions have concrete limitations, in that they relate to some predefined source (e.g., a provided context or a specific field, like medicine).

A machine comprehends a passage of text if, for any question regarding the text that can be answered correctly by a majority of native speakers, that machine can provide a string which those speakers would agree both answers that question, and does not contain information irrelevant to that question [1].

Here Christopher, etc. defined machine comprehension in terms of Question Answering in its most general form.

2. Motivation

The goal of this project is to learn deep learning models on Question Answering systems, and to explore SQuAD dataset and implement the most recent google-research BERT model.

3. Related work

The processing of a QA system may broadly have three stages, i.e., question analysis: parsing, question classification and query reformulation; document analysis: extract candidate document, identify answers; and answer analysis: extract candidate answers and rank the best one [2].

Question Processing receives the input from the user, a question in natural language, to analyze and classify it. The analysis is to find out the type of question, meaning the focus of the question. This is necessary to avoid ambiguities in the answer (Malik et al., 2013) [3].

Different from question processing that is execute on every question asked by the user, Document Processing has as its main feature the selection of a set of relevant documents and the extraction of a set of paragraphs depending on the focus of the question or text understanding throw natural language processing.

The Answer Processing is the most challenging task on a Question Answering system. This module uses extraction techniques on the result of the Document Processing module to present an answer (Bhoir and Potey, 2014) [4].

Besides the main architecture, QA systems can be defined by the paradigm each one implements:

(1) Information Retrieval QA:

Usage of search engines to retrieve answers and then apply filters and ranking on the recovered passage.

(2) Natural Language Processing QA:

Usage of linguistic intuitions and machine learning methods to extract answers from retrieved snippet.

(3) Knowledge Base QA:

Find answers from structured data source (a knowledge base) instead of unstructured text. Standard database queries are used in replacement of word-based searches (Yang et al., 2015) [5]. This paradigm make use of structured data, such as ontology. An ontology describes a conceptual representation of concepts and their relationships within a specific domain. Ontology can be considered as a knowledge base which has a more sophisticated form than a relational database (Abdi et al., 2016) [6]. To execute queries in order to retrieve knowledge from the ontology, structured languages are proposed and one of them is SPARQL.

(4) Hybrid QA:

High performance QA systems make use of as many types of resources as possible, especially with the prevailing popularity of modern search engines and enriching community-contributed knowledge on the web. A hybrid approach is the combination of IR QA, NLP QA and KB QA. The main example of this paradigm is IBM Watson (Ferrucci et al., 2013) [7].

Question answering task combines techniques from artificial intelligence, natural language processing, statistical analysis, pattern matching, information retrieval, and information extraction.

4. SQuAD Data

To support the development of the state-of-art Machine Learning (ML) models for AQA and MC, a number of large datasets were created. These include Stanford's SQuAD for automated question answering, MS Marco for real-world question answering, Trivia QA for complex compositional answers and multi-sentence reasoning, CNN/Daily Main and Children's Book Test dataset for cloze-style reading comprehension, and others.

Stanford Question Answering Dataset (SQuAD) is a new reading comprehension dataset, consisting of questions posed by crowd workers on a set of Wikipedia articles, where the answer to every question is a segment of text, or span, from the corresponding reading passage, or the question might be unanswerable [8]. With 100,000+ question-answer pairs on 500+ articles, SQuAD is significantly larger than previous reading comprehension dataset.

Example of context, question and answer on SQuAD:

Context:

"Architecturally, the school has a Catholic character. Atop the Main Building's gold dome is a golden statue of the Virgin Mary. Immediately in front of the Main Building and facing it, is a copper statue of Christ with arms upraised with the legend \"Venite Ad Me Omnes\". Next to the Main Building is the Basilica of the Sacred Heart. Immediately behind the basilica is the Grotto, a Marian place of prayer and reflection. It is a replica of the grotto at Lourdes, France where the Virgin Mary reputedly appeared to Saint Bernadette Soubirous in 1858. At the end of the main drive (and in a direct line that connects through 3 statues and the Gold Dome), is a simple, modern stone statue of Mary."

Question:

To whom did the Virgin Mary allegedly appear in 1858 in Lourdes France?

Answer:

Saint Bernadette Soubirous

The public SQuAD Dataset is in json file, and its structure is as below.

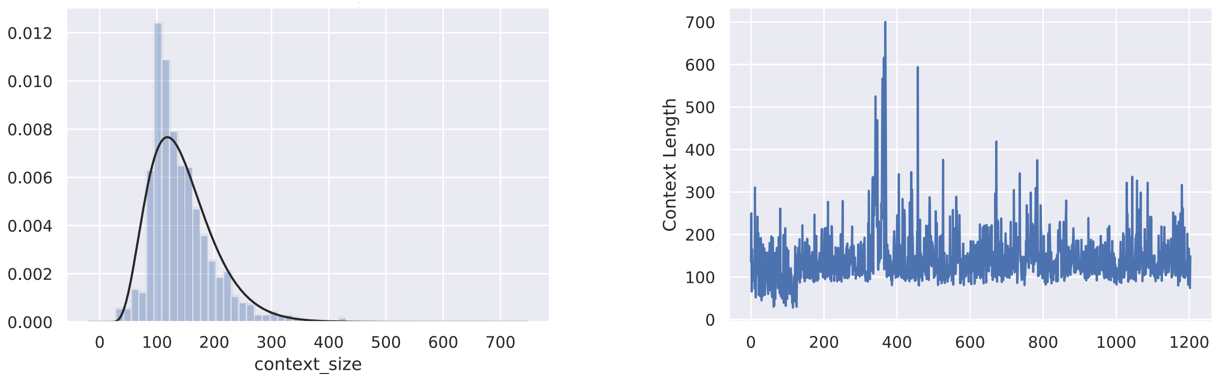
```
structure:
  article (dict)
   ├──── title (str)
   ├──── paragraphs (dict)
         ├──── context (str)
         └──── qas (dict)
              ├──── question (str)
              ├──── id (str)
              └──── answers (dict)
                   ├──── answer_start (integer)
                   └──── text (str)
  ...
   └─── paragraphs n
      ├─── context
      └─── qas
```

SQuAD 2.0 combines the 100,000 questions in SQuAD 1.1 with over 50,000 unanswerable questions written adversarially by crow workers to look similar to answerable ones.

The SQuAD dataset is a closed dataset meaning that the answer to a question is always a part of the context and also a continuous span of context. So the problem of finding an answer can be simplified to finding the start index and the end index of the context that corresponds to the answers.

5. Data Exploration

SQuAD 2.0 Dataset is explored by word size, i.e. length.

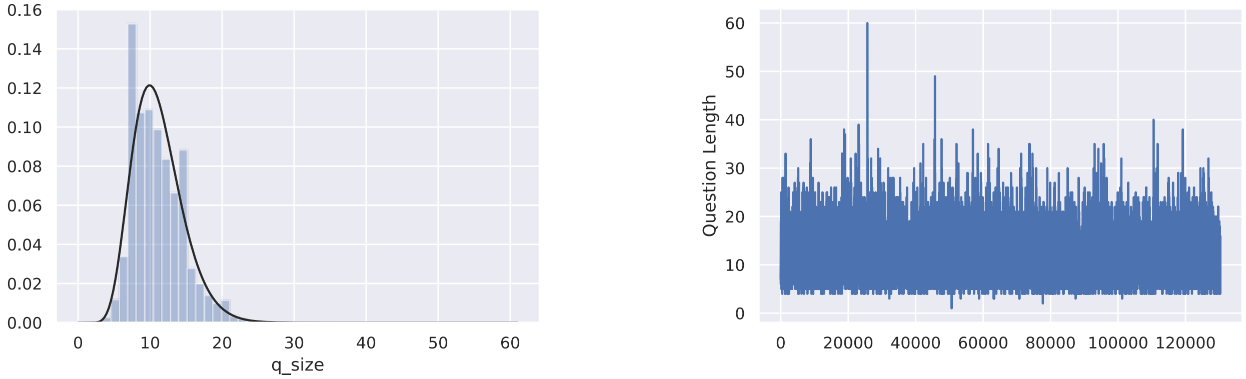


(a) (b)

Figure 1: (a) Train Dataset Context Length Density, (b) Train Dataset Context Length Distribution by tuples

Context:

The majority of context are 100 to 200 word-size long, and we could see that the context located around 400 index speared highly from Figure 1.

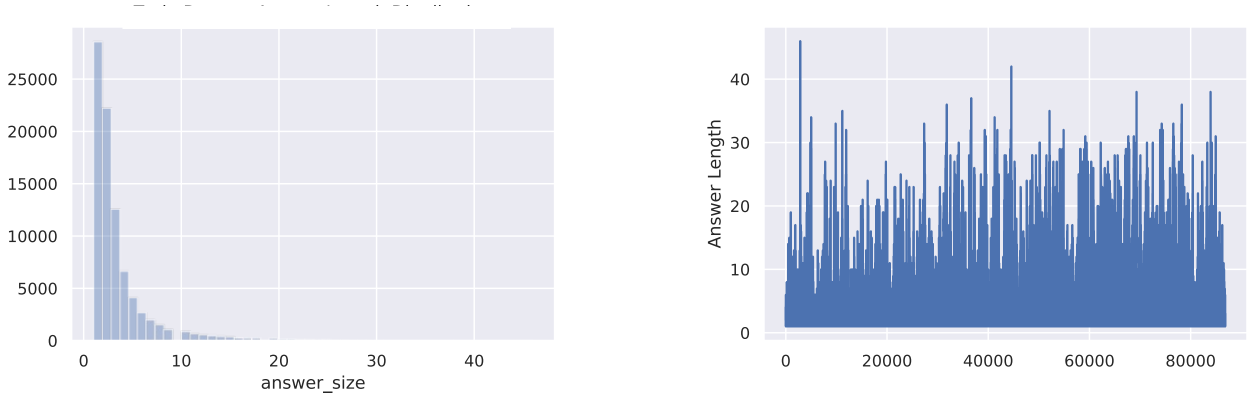


(a) (b)

Figure 2: (a) Train Dataset Question Length Density, (b) Train Dataset Question Length Distribution by tuples

Question:

From Figure 2, we noticed that more than 90% of questions are less than 20 word-size long, and most of tuples are similar long since the standard deviation of word size is not large.

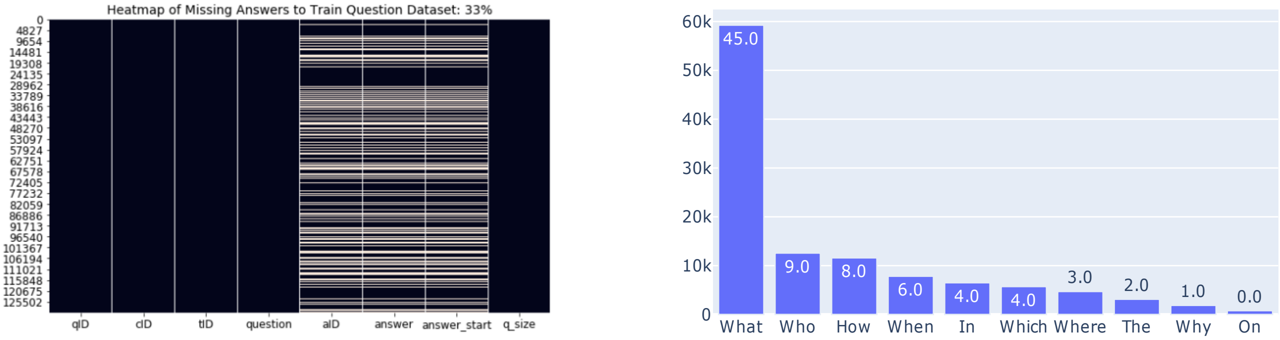


(a) (b)

Figure 3: (a) Train Dataset Answer Length Density, (b) Train Dataset Answer Length Distribution by tuples

Answer:

From Figure 3, we could observe that most of answers consists of two or three words. Around 90% of answers contain 10 words. The length of answers is a bit various than contexts and questions since some answers carry more words than others.



(a) (b)

Figure 4: (a)Tuples of Question without an answer Distribution, (b) First Word of Question Frequency Distribution

We also gave a visualization of missing answer distribution on tuples. Besides we ordered the most frequent asked first word of questions, and it turns out that [What, Who, How, When, In, Which, Where, The, Why, On] account for 82% of total.

6. BERT

BERT, or **B**idirectional **E**ncoder **R**epresentations from **T**ransformers is a recent paper published by researchers at Google AI Language [9]. It is a new method of pre-training language representations which obtains state-of-the-art results on a wide array of NLP tasks, including Question Answering (SQuAD), Natural Language Inference (MNLI), and others.

Unlike recent language representation models (Peters et al., 2018a) [10], BERT is designed to pretrain deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers. As a result, the pre-trained BERT model can be fine-tuned with just one additional output layer to create state-of-art models for a wide range of tasks without substantial task-specific architecture modifications. In the paper, the researchers detail a novel technique named Masked LM (MLM) which allows bidirectional training in models in which it was previously impossible.

6.1 Transformer

BERT makes use of Transformer, an attention mechanism that learns contextual relations between words (or sub-words) in a text. Transformer includes two separate mechanisms – an encoder that reads the text input and the decoder that produces a prediction for the task. Since BERT's goal is to generate a language model, only the encoder mechanism is necessary. The detailed workings of Transformer could be referred to the paper by Google.

As opposed to directional models, which read the text input sequentially (left-to-right or right-to-left), the Transformer encoder reads the entire sequence of words at once. Bidirectional characteristic allows BERT to learn the context of a word based on all of its surroundings (left and right of the word).

Figure 5 illustrates a high-level description of the Transformer encoder. The input is a sequence of tokens, which are first embedded into vectors and then processed in the neural network. The output is a sequence of vectors of size H, in which each vector corresponds to an input token with the same index.

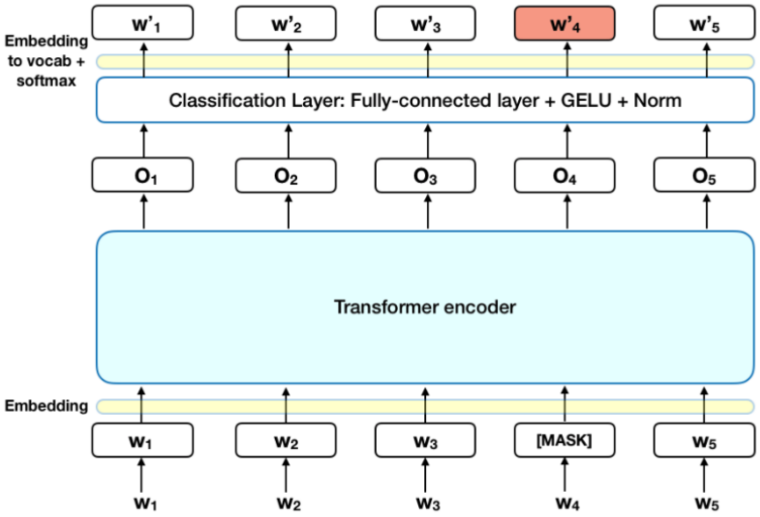


Figure 5. Architecture of Transformer encoder
(Source: Towards Data Science)

When training the language models, many models predict the next word in a sequence (e.g. "The child came home from __"), but due to its directional characteristic, BERT inherently limits context learning. To resolve the problem, BERT uses two training strategies: Whole Word Masking and Next Sentence Prediction.

6.2 Masked Language Model (Masked LM / MLM)

Before feeding word sequences into BERT, 15% of the words in each sequence are replaced with a [MASK] token. The model then attempts to predict the original value of the masked words, based on the context provided by the other, non-masked, words in the sequence. The prediction of the output words requires:

(1) Adding a classification layer on top of the encoder output.

(2) Multiplying the output vectors by the embedding matrix, transforming them into the vocabulary dimension.

(3) Calculating the probability of each word in the vocabulary with softmax.

Noticed: The BERT loss function only considers the prediction masked values and ignores the prediction of the non-masked words. As a consequence, the model converges slower than directional models, which is a offset by its increased context awareness.

6.3 Next Sentence Prediction (NSP)

Many important downstream tasks such as Question Answering (QA) and Natural Language Inference (NLI) are based on understanding the relationship between two sentences, which is not directly captured by language modeling. In order to train a model that understands sentence relationships, the paper, pre-train for a binarized next sentence prediction task that can be trivially generated from any monolingual corpus. Specifically, when choosing the sentences, A and B for each pretraining example, 50% of the time B is the actual next sentence that follows A (labeled as IsNext), and 50% of the time it is a random sentence from the corpus (labeled as NotNext).

To help the model distinguish between the two sentences in training, the input is processed in the following ways before entering the model:

(1) A [CLS] token is inserted at the beginning of the first sentence and a [SEP] token is inserted at the end of each sentence.

(2) A sentence embedding indicating Sentence A or Sentence B is added to each token. Sentence embeddings are similar in concept to token embeddings with a vocabulary of 2.

(3) A positional embedding is added to each token to indicate its position in the sequence. The concept and implementation of positional embedding are presented in the Transformer paper.

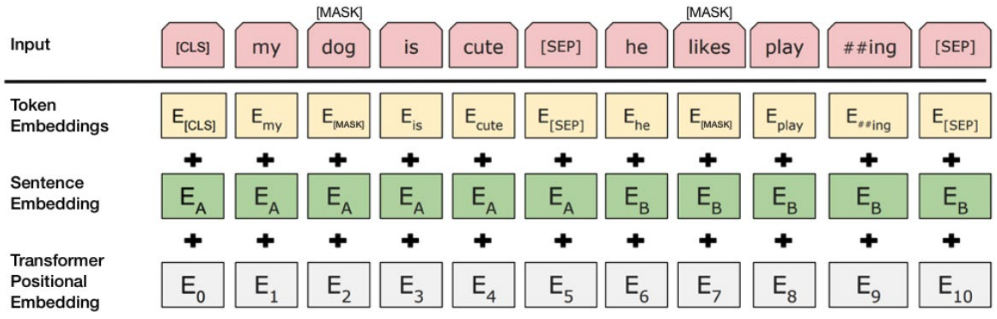

Input
[CLS]
my
[MASK]
dog
is
cute
[SEP]
he
[MASK]
likes
play
##ing
[SEP]
Token Embeddings
$E_{[CLS]}$ E_{my} $E_{[MASK]}$ E_{is} E_{cute} $E_{[SEP]}$ E_{he} $E_{[MASK]}$ E_{play} $E_{\#\#ing}$ $E_{[SEP]}$
Sentence Embedding
E_A E_A E_A E_A E_A E_A E_B E_B E_B E_B E_B
Transformer Positional Embedding
E_0 E_1 E_2 E_3 E_4 E_5 E_6 E_7 E_8 E_9 E_{10}

Figure 6. BERT input representation. The input embeddings are the sum of the token embeddings, the segmentation embeddings and the position embeddings. (Source: BERT paper)

When training the BERT model, Masked LM and Next Sentence Prediction are trained together, with the goal of minimizing the combined loss function of the two strategies.

7. BERT Implementation & Evaluation

7.1 Model Implementation

According to BERT paper, using BERT has two stages: Pre-training and fine-tuning.

Pre-training:

It is fairly expensive (four days on 4 to 16 Cloud TPUs) but is a one-time procedure for each language (current models are English-only, but multilingual models will be released in the near future). We are releasing a number of pre-trained models from the paper which were pre-trained at Google. Most NLP researchers will never need to pre-train their own model from scratch.

Fine-tuning:

It is inexpensive. All of the results in the paper can be replicated in at most 1 hour on a single Cloud TPU, or a few hours on a GPU, starting from the exact same pre-trained model. SQuAD, for example, can be trained in around 30 minutes on a single Cloud TPU to achieve a Dev F1 score of 91.0%, which is the single system state-of-the-art.

Here fine-tuned model **bert uncased_L-12_H-768_A-12** is used. Code is based on BERT repository and modified according.

We used Colab **TPU** and **GS bucket** for data storage, and the environment set up steps have been described in the jupyter notebook.

We have three major functions:

(1) read_squad_examples: This function takes json file and turns it into SquadExample, which is a class with data attributes.

(2) get_final_text: This function is used to predictions of binary number to original text. It is internally used in write_predictions.

(3) write_predictions: This function writes prediction data to json file, and log-odds of null if needed. The output prediction.json is a dict of {{'question_id', 'predicted_answer'}}.

(4) convert_examples_to_features: This function converts train examples got from read_squad_examples to features that can be fed to BERT model. It returns all features in InputFeatures class object.

7.2 Model Evaluation

F1 scores are used for evaluation.

F1 score computes the average word overlap between predicted and correct answers.

F 1 = 2 * Precision * Recall / (Precision + Recall)

We used SQuAD 1.1 Dataset for training, and with fine-tuned model checkpoints, we got a F1 score of **87.98%** on dev-v1.1 test dataset.

Here one piece of test example is listed as below:

The highlighted part is where the predicted answer differs from the given answer.

```
Test Sample:
-------------------------------------------------------------------
Context:
Super Bowl 50 was an American football game to determine the champion of the National
Football League (NFL) for the 2015 season. The American Football Conference (AFC)
champion Denver Broncos defeated the National Football Conference (NFC) champion
Carolina Panthers 24–10 to earn their third Super Bowl title. The game was played on
February 7, 2016, at Levi's Stadium in the San Francisco Bay Area at Santa Clara,
California. As this was the 50th Super Bowl, the league emphasized the "golden
anniversary" with various gold-themed initiatives, as well as temporarily suspending the
tradition of naming each Super Bowl game with Roman numerals (under which the game
would have been known as "Super Bowl L"), so that the logo could prominently feature the
Arabic numerals 50.
-------------------------------------------------------------------
Question:
Which NFL team represented the AFC at Super Bowl 50?
Given answer: Denver Broncos
Predict answer: Denver Broncos
-------------------------------------------------------------------
Question:
Which NFL team represented the NFC at Super Bowl 50?
Given answer: Carolina Panthers
Predict answer: Carolina Panthers
-------------------------------------------------------------------
Question:
Where did Super Bowl 50 take place?
Given answer: Levi's Stadium
Predict answer: Levi's Stadium in the San Francisco Bay Area at Santa Clara, California
```

8. Summary

This project visualized dada analysis on SQuAD 2.0 Dataset and implemented BERT fine-tuned model on SQuAD Dataset and achieved a relatively good result of F1 score of 87.98% on dev-v1.1 test dataset. The results show that BERT model did well in closed Question Answering problems. More work could be done if we combine other NLP teniques.

9. Reference

[1] Burges, C.J., 2013. Towards the machine comprehension of text: An essay. *TechReport: MSR-TR-2013-125*.

[2] Dwivedi, S.K. and Singh, V., 2013. Research and reviews in question answering system. *Procedia Technology*, *10*, pp.417-424.

[3] Malik, N., Sharan, A. and Biswas, P., 2013, December. Domain knowledge enriched framework for restricted domain question answering system. In *2013 IEEE International Conference on Computational Intelligence and Computing Research* (pp. 1-7). IEEE.

[4] Bhoir, V. and Potey, M.A., 2014, February. Question answering system: A heuristic approach. In *The Fifth International Conference on the Applications of Digital Information and Web Technologies (ICADIWT 2014)* (pp. 165-170). IEEE.

[5] Yang, M.C., Lee, D.G., Park, S.Y. and Rim, H.C., 2015. Knowledge-based question answering using the semantic embedding space. *Expert Systems with Applications*, *42*(23), pp.9086-9104.

[6] Abdi, A., Idris, N. and Ahmad, Z., 2018. QAPD: an ontology-based question answering system in the physics domain. *Soft Computing*, *22*(1), pp.213-230.

[7] Ferrucci, D.A., 2012. Introduction to “this is watson”. *IBM Journal of Research and Development*, *56*(3.4), pp.1-1.

[8] Rajpurkar, P., Zhang, J., Lopyrev, K. and Liang, P., 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.

[9] Devlin, J., Chang, M.W., Lee, K. and Toutanova, K., 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

[10] Peters, M.E., Ammar, W., Bhagavatula, C. and Power, R., 2017. Semi-supervised sequence tagging with bidirectional language models. *arXiv preprint arXiv:1705.00108*。

[11] https://rajpurkar.github.io/SQuAD-explorer/