Question and Answering Using BERT

Suman Karanjit, Computer Science Major

Minnesota State University Moorhead

Link to GitHub

https://github.com/skaranjit/BertQA

Table of Contents

Abstract

There are many machine learning algorithms that tries to solve the question answering using the natural language. In old day, the bag of words was popular amongst others which tries to answer the question that are pre-defined by the developers. Using this method, developers have to spend a lot of time writing the questions and answer for the particular questions. This method was very useful for the chat bots but was not able to answer the questions for the huge database. Modern Natural Language Processor are dominated by the models called transformers. Using these transformers library, the Bert QA model was introduced by HuggingFace which reads through the text context provided by the user and tries to answer the questions related to that text context. This model has been promising in answering the complex question from a large document. For example, if one company have a report regarding their financial years that been fed through the model, user can just ask question regarding the certain year or the profits they made for particular year. And without scrolling through the documents the answer can be found with the matter of seconds.

Introduction

The Question Answering have been a very crucial part of any language study. The term Question Answering here comes from the reading comprehensive where the reader is given certain paragraphs to read and answer some questions related to those paragraphs. Using this concept, there are many models that are trained to do so using machine learning algorithms. They can understand the text context and answer the question related to that context. They can answer using the natural language that we are used to. It not only answers but also tries to correctly understand the sentiments of questions and look for the particular text or sentence it is looking for. To achieve this, I will be using the “Bert Model from HugginFace Transformers” which are available from the HugginFace websites to download and use. I will also be using different libraries in python to achieve this.

SQuAD

SQUAD stands for Stanford Question Answering Dataset which is a reading comprehension dataset. It is collection of question 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 is. SQuAD 1.1 was used to fine tune the model which contains 100,000+ question-answer pairs on 500+ articles.

BERT Explained

What is BERT?

BERT, Bi-directional Encoder Representation from Transformer, is language model by Google which can be used for Natural Language Processing (NLP) tasks. BERT is a transfer learning model meaning that it is being trained for one task but can be reuse for another task. BERT is a trained Transformer Encoder stack, with twelve in the Base version, and twenty-four in the Large version. There is many BERT provided by Hugginface for multiple tasks like, Summarization, Fill-Mask, Question-Answering etc. For the purpose of this paper, I will discuss about the BERT Question-Answering where the model is feed with the context and a question which will generate the answer based on the context.

Architecture

BERT is a trained Transformer Encoder Stack with twelve in the Base version and twenty-four in Large Version. As in Fig 1, I have included the Bert model with the embedding layer at the top, Encoder Layer (included just one but will repeat same 12 times on Base and 24

times on Large), and end layer with Linear Layer.Before the first layer of encoder, the positional

```
BertForQuestionAnswering(
 (bert): BertModel(
  (embeddings): BertEmbeddings(
   (word_embeddings): Embedding(30522, 768, padding_idx=0)
   (position_embeddings): Embedding(512, 768)
   (token_type_embeddings): Embedding(2, 768)
   (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
   (dropout): Dropout(p=0.1, inplace=False)
  )
  (encoder): BertEncoder(
   (layer): ModuleList(
    (0): BertLayer(
     (attention): BertAttention(
      (self): BertSelfAttention(
       (query): Linear(in_features=768, out_features=768, bias=True)
       (key): Linear(in_features=768, out_features=768, bias=True)
       (value): Linear(in_features=768, out_features=768, bias=True)
       (dropout): Dropout(p=0.1, inplace=False)
      )
      (output): BertSelfOutput(
       (dense): Linear(in_features=768, out_features=768, bias=True)
       (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
       (dropout): Dropout(p=0.1, inplace=False)
      )
     )
     (intermediate): BertIntermediate(
      (dense): Linear(in_features=768, out_features=3072, bias=True)
     )
     (output): BertOutput(
      (dense): Linear(in_features=3072, out_features=768, bias=True)
      (LayerNorm): LayerNorm((768,), eps=1e-12, elementwise_affine=True)
      (dropout): Dropout(p=0.1, inplace=False)
     )
    )
    )
   )
  )
 )
 (qa_outputs): Linear(in_features=768, out_features=2, bias=True)
)
```

Figure 1

embedding is added. The output of the last layer is fed into layer containing linear layer and SoftMax and generates the predicted probability of the start of the answer and the end position of the answer.

Input Processing

For this experiment, we use publicly available Bert-model(bert-large-uncased-whole-word-masking-finetuned-squad) that has already been pre-trained and fine-tuned on SQUAD V1 dataset which consists of 100,000+ questions. For the question answering task, Bert takes the input question and passage (context) as a single packed sequence. The input embeddings are the

sum of the token embeddings and the segment embeddings. Below is the process of input processing.

Token embeddings: A [CLS] token is added to the input word tokens at the beginning of the question and a [SEP] token is inserted at the end of both the question and the paragraph.

Segment embeddings: A marker indicating Sentence A or Sentence B is added to each token. This allows the model to distinguish between sentences. In the below example, all tokens marked as A belong to the question, and those marked as B belong to the paragraph.

Here is an example input processing that includes passage, question, Tokens, segment_ids and Input ids. Note that the segment ids only include 0 and 1 which represents question and a passage text respectively.

```
Passage:
John is a 10 year old boy. He is the son of Robert Smith.  Elizabeth Davis is
Robert's wife. She teaches at UC Berkeley. Sophia Smith is Elizabeth's daughter.
She studies at UC Davis

Question:
Who is John's sister?

Tokens:
 ['[CLS]', 'who', 'is', 'john', "'", 's', 'sister', '?', '[SEP]', 'john', 'is', 'a', '10', 'year'
, 'old', 'boy', '.', 'he', 'is', 'the', 'son', 'of', 'robert', 'smith', '.', 'elizabeth', 'davis'
, 'is', 'robert', "'", 's', 'wife', '.', 'she', 'teaches', 'at', 'uc', 'berkeley', '.', 'sophia',
'smith', 'is', 'elizabeth', "'", 's', 'daughter', '.', 'she', 'studies', 'at', 'uc', 'davis', '[S
EP]']

Segment Ids:
 [0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]

 Input Ids:
 [101, 2040, 2003, 2198, 1005, 1055, 2905, 1029, 102, 2198, 2003, 1037, 2184, 2095, 2214, 2879, 1
012, 2002, 2003, 1996, 2365, 1997, 2728, 3044, 1012, 3870, 4482, 2003, 2728, 1005, 1055, 2564, 10
12, 2016, 12011, 2012, 15384, 8256, 1012, 9665, 3044, 2003, 3870, 1005, 1055, 2684, 1012, 2016, 2
913, 2012, 15384, 4482, 102]
```

Figure 2

Getting Answer

After passing the input tokens through layers of transformers, it generates the word embedding and also introduces a start vector.The probability of each word being the start-word is calculated by taking a dot product between the final embedding of the word and the start vector and applies SoftMax over all the words. The word with the highest probability value is considered the start of the answer.

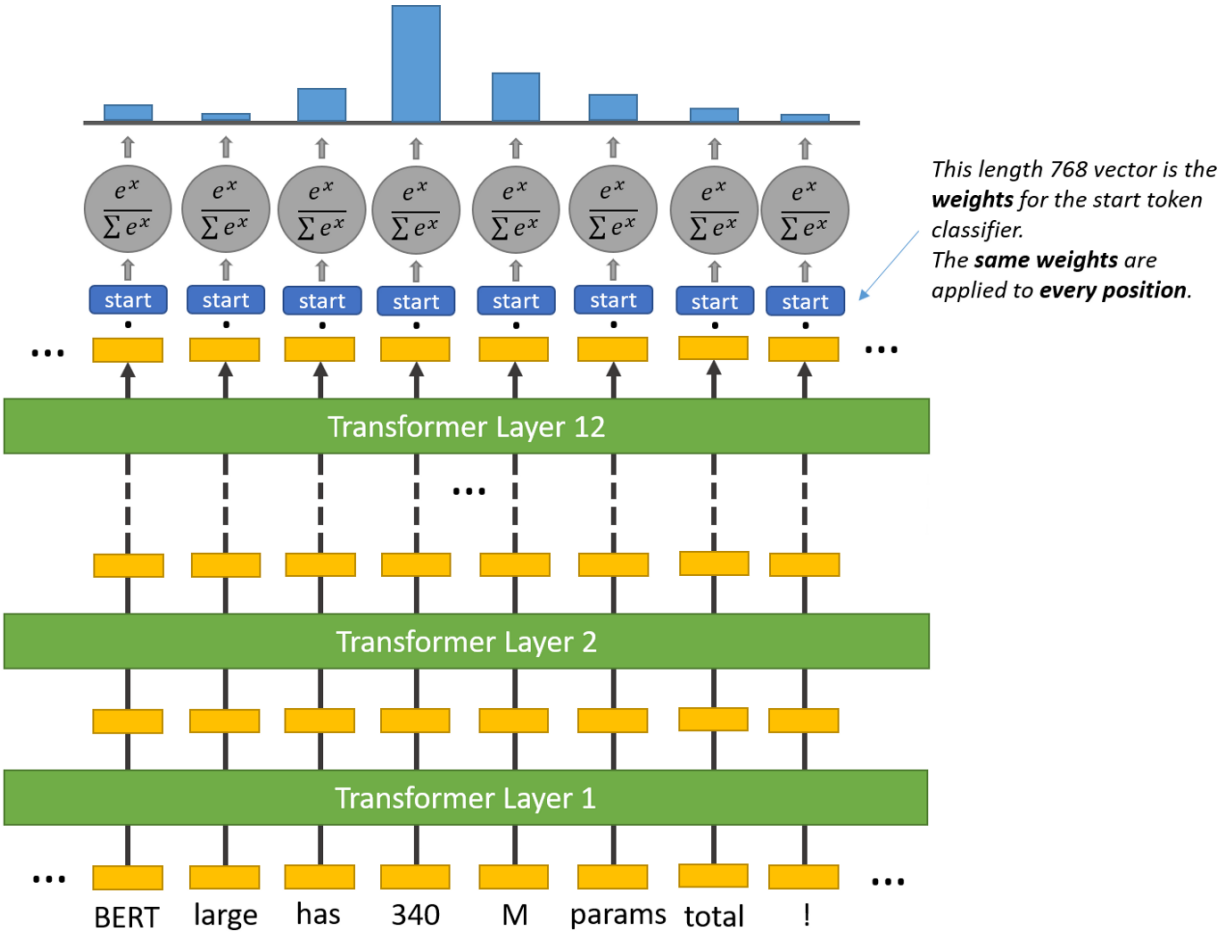


Figure 3

It repeats the same process to find the end-word of an answer.

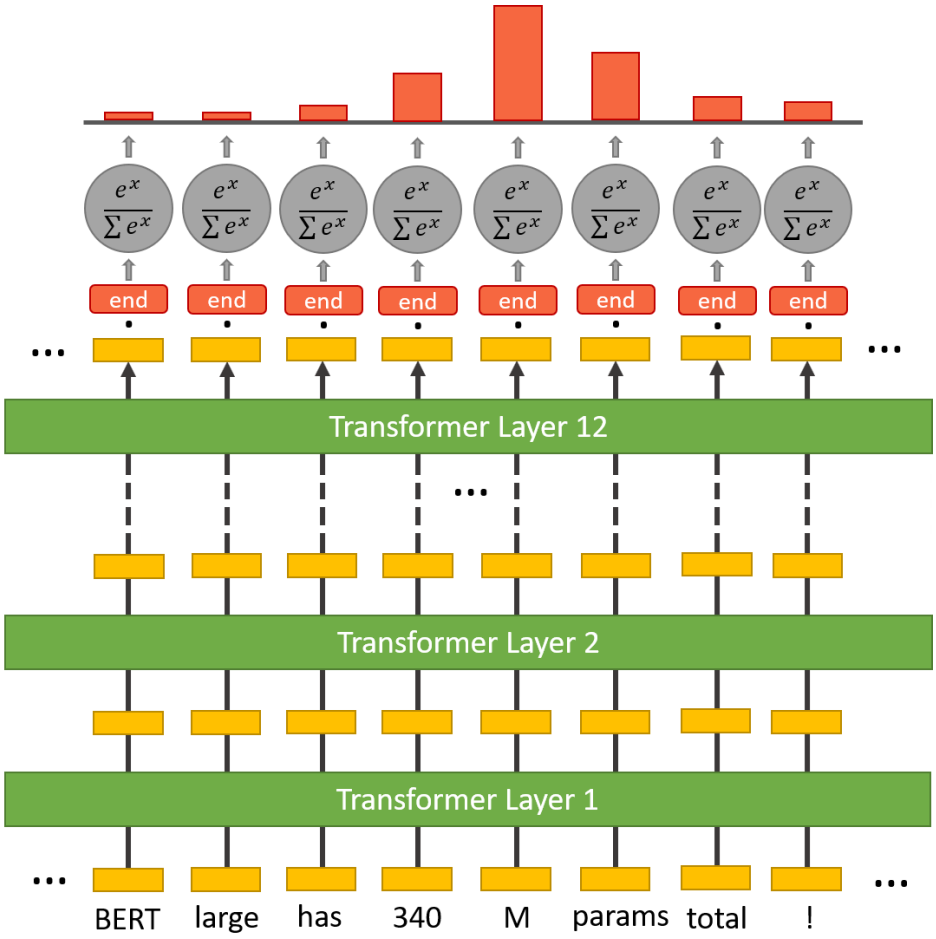


Figure 4

Once we find the start and end word, we can generate a full answer text from the context by grabbing the start word to end word.

Setting up the environment.

The link above to the git hub have all necessary packages that needs to be installed for the project to work. Once python 3.6 has been installed, we can follow below code to install the required packages as well.

```
# pip install -r requirement.txt
```

This will install all the required packages for the project. There are 2 main python modules that I have written that are in the heart of this project.

1. loadModel.py : This module will load the BERT model process inputs and generates outputs. This module has a class QAPipe which we will use to interact with the BERT model.
2. create_dataset.py: This module will create a dataset from either text or article from web address. It has a class named Create_DS which is used to generate the clean context from either a text passage or article from the internet.

Also, I have included the jupyter notebook that anyone can use to better understand the model and how it works.

THE PROCESS

Firstly, we load the modules as shown on the picture “Figure 5”:

```
[ ]:
import create_dataset
from loadModel import *
import textwrap
```

Figure 5

After importing the modules, we can start using as shown below on Figure 6.

```
[ ]: p = create_dataset.Create_DS()
     p.loadTxt("John is a 10 year old boy. He is the son of Robert Smith.  Elizabeth Davis is Robert's wife. She
     model = QAPipe(p.ds)
     question = "Which college does John's sister attend?"
     answer_start_index,answer_end_index,start_token_score,end_token_score,s_Scores,e_Scores,answer=model.get_outp
```

Let check the answer

```
[6]: wrapper = textwrap.TextWrapper(width=80)
     print(wrapper.fill(p.ds)+"\n")

     print("Question: "+question)
```

```
John is a 10 year old boy. He is the son of Robert Smith.  Elizabeth Davis is
Robert's wife. She teaches at UC Berkeley. Sophia Smith is Elizabeth's daughter.
She studies at UC Davis

Question: Which college does John's sister attend?
```

```
[7]: print("Answer : " + answer)
```

```
Answer : uc davis
```

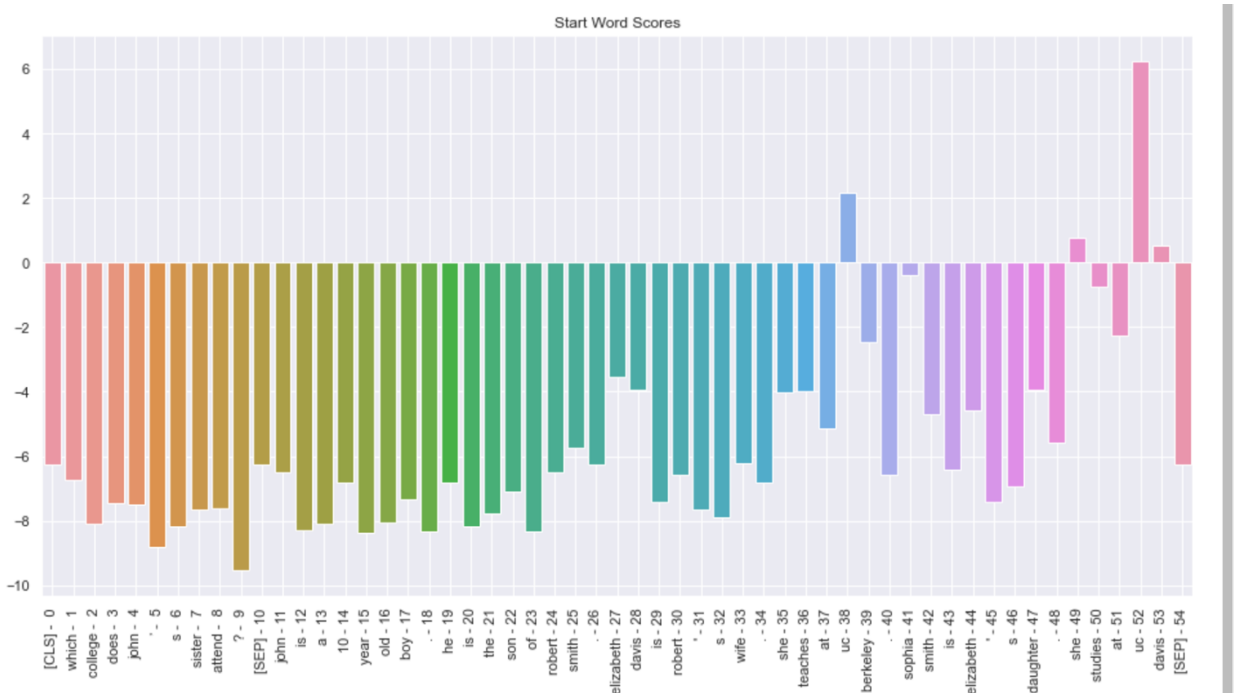
Figure 6

As shown on Figure 6, we use the passage text: "John is a 10-year-old boy. He is the son of Robert Smith. Elizabeth Davis is Robert's wife. She teaches at UC Berkeley. Sophia Smith is Elizabeth's daughter. She studies at UC Davis". We have model as QAPipe(p.ds) after loading the text to our p=create_dataset which will clean the text. We get the answer by calling one of the methods called get_output by feeding our question. This method will return the answer including the start and end index, start and end token score and start and end word score.

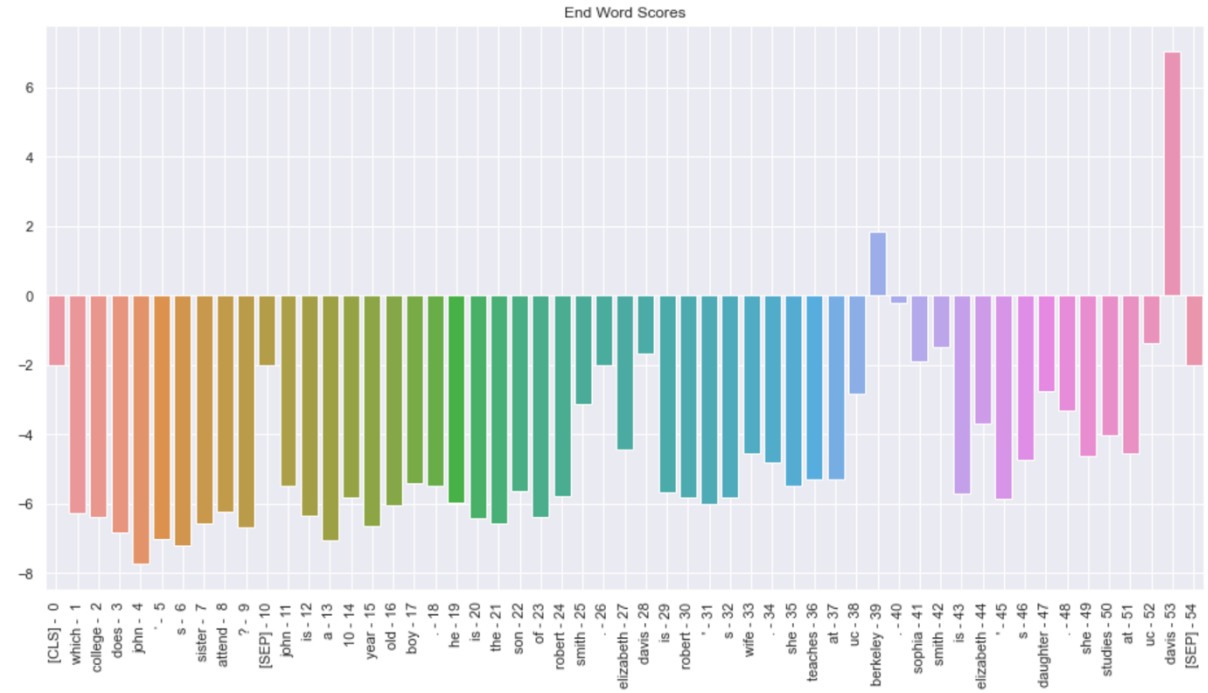
Visualization

Below Figure 7 shows the visualization of the start word and the end word weight so that the model can select the best possible answer.

Start Word Weight



End Word Weight



Conclusion

In addition to question answering task, BERT can also understand the language structure and handle dependencies across sentences. It can apply simple logic to answer the questions. As mention on above example, BERT was able to answer the question “Which college does John’s sister attend?” correctly without any mention of john’s sister in the context “John is a 10-year-old boy. He is the son of Robert Smith. Elizabeth Davis is Robert's wife. She teaches at UC Berkeley. Sophia Smith is Elizabeth's daughter. She studies at UC Davis”. Also note that the longer passage can be passed through the model but if the length of the question and passage exceeds 512 tokens, the code will automatically truncate the extra part. These models can be used in very useful environment like chat-bot, querying the long business reports, etc.

Works Cited

1. "Python 3.9.1 Documentation." Accessed February 3, 2021. https://docs.python.org/3/.

   Easy way to go access the information from this source https://docs.python.org/3/ . It includes all the libraries documentation as well as the installation. The source helps me to setup the environment for the work.

2. Cdqa-Suite. "Cdqa-Suite/CdQA." GitHub. Accessed February 3, 2021. https://github.com/cdqa-suite/cdQA.

   Easy way to access the source above is to access through https://github.com/cdqa-suite/cdQA. The above source is the git hub repo for the codes that I will make a base on my project. It is not currently maintained but provide the baseline for my work. It does provide the model.

3. Swalin, Alvira. "Building a Question-Answering System from Scratch- Part 1." Medium. Towards Data Science, June 1, 2018. https://towardsdatascience.com/building-a-question-answering-system-part-1-9388aadff507.

   Easy way to access the source is through the link: https://towardsdatascience.com/building-a-question-answering-system-part-1-9388aadff507 . This source provides a good detail of the machine learning algorithms and training module. Although it uses infersent model to train and evaluate, I can use this source to gain more knowledge for my work which I will be using bert model.

4. Horev, Rani. “BERT Explained: State of the Art Language Model for NLP.” Medium. Towards Data Science, November 17, 2018. https://towardsdatascience.com/bert-explained-state-of-the-art-language-model-for-nlp-f8b21a9b6270.

   Easy way to access the source is by: https://towardsdatascience.com/bert-explained-state-of-the-art-language-model-for-nlp-f8b21a9b6270

   This article lays the very good knowledge about the usage of bert model, training, fitting and using it to process the Natural Language. I will also be using same bert model and same squad data set in my project to achieve QA model.

5. “Pretrained Models¶.” Pretrained models - transformers 4.2.0 documentation. Accessed February 3, 2021. https://huggingface.co/transformers/pretrained_models.html.

   Easy way to access the source is by: https://huggingface.co/transformers/pretrained_models.html

   This source provides the libraries and models for the machine learning of different scenario on Natural language processing. The one I will be using is the question answering model.

6. “SpaCy · Industrial-Strength Natural Language Processing in Python.” · Industrial-strength Natural Language Processing in Python. Accessed February 3, 2021. https://spacy.io/.

   Easy way to access the source: https://spacy.io

   This source provides the information about spacy libraries that will be used in python for Natural Language Processing. It has many useful libraries that can be used from tokenization of the sentence or downloading the language model such as English or

even the pulling out the root word from the sentence. It also can help break down the sentence composition as well.

7. “NumPy v1.20 Manual.” Overview - NumPy v1.20 Manual. Accessed February 3, 2021. https://numpy.org/doc/stable/.

   Easy way to access the source: https://numpy.org/doc/stable/

   This source provides the information regarding numpy which is used to create a multi-dimensional array in python which will be used for machine learning.

8. **“**Natural Language Toolkit — NLTK 3.5 Documentation.” Accessed February 11, 2021. https://www.nltk.org.

   Easy way to access the source: https://www.nltk.org.

   The source provides the information regarding nltk library. It also provides the documentation on how to use the nltk library. The nltk library is the one of most used libraries while processing natural language like sentence embedding, creating tokens etc.

   This source can be directly relate to my project as I am using nltk libraries to embed sentence from the paragraph.

9. Wei, Jerry. “The Quick Guide to SQuAD. All the Basic Information You Need To... | by Jerry Wei | Towards Data Science.” Medium. Towards Data Science, October 8, 2020. https://towardsdatascience.com/the-quick-guide-to-squad-cae08047ebee.

Easy way to access: https://towardsdatascience.com/the-quick-guide-to-squad-cae08047ebee

The source provides the information regarding Squad Datasets which is a huge data sets for question answering and how it can be used in machine learning. It also provides the insight of its history and uses.

The squad data sets are used to train and evaluate question answering models. The squad data sets are huge collection of paragraphs, and question & answer related to that particular paragraph. The squad data set was used to train my model as well.

10. Michel Kana, P. (2021, January 07). BERT NLP - how to build a question Answering Bot. Retrieved April 16, 2021, from https://towardsdatascience.com/bert-nlp-how-to-build-a-question-answering-bot-98b1d1594d7b

    This source provides a detailed explanation on how the bert works and I did took some pictures from this site to use in this paper.

11. Anonymous (n.d.). How Does BERT Answer Questions? 59 60 3 A Layer-Wise Analysis of Transformer Representations. Retrieved April 16, 2021, from https://openreview.net/pdf?id=SygMXE2vAE

    This paper provides the analysis of each layer in the Bert model.I find it very useful for this paper although the author is anonymous, the work put on this paper looks genuine. The writer has explained in detail about the Hidden Layer.

12. Project Jupyter | Home. “Project Jupyter | Home.” Accessed February 11, 2021. https://jupyter.org.

Easy way to access: https://jupyter.org.

The source provides information regarding the jupyter notebook which is a web-based interactive for python. The source provides the instructions on installation and uses. The Jupyter notebook provides the web-interactive shell in which we can work on for machine learning. I can use this notebook to run the code separately so that I do not need to run that code again to move to next step until necessary. It saves a lot of time comparing to running the program. It also displays the results in the web itself so that the analysis is easier.

13. W3Schools Online Web Tutorials. “Python Machine Learning.” Accessed February 11, 2021. https://www.w3schools.com/python/python_ml_getting_started.asp. Easy way to access the source:

    https://www.w3schools.com/python/python_ml_getting_started.asp

    The source provides the learning interface for the machine learning techniques and concepts. The source also does provide the method that the data can be interpreted using the techniques such as linear regression, etc.

    This source will help me learn the way of machine learning that can be used to complete my project.