**CPSC - 8430**

**HomeWork 3 Report**

**Task: Extractive Question Answering**

**Sravani Pati**

**Github Link:** [**https://github.com/sravani919/DeepLearning-Hw3\_SravaniPati.git**](https://github.com/sravani919/DeepLearning-Hw3_SravaniPati.git)

**Introduction:**

A speech-based version of SpokenSquad dataset contains questions and responses about a range of subjects. The goal of this dataset is to make it easier to create question-answering models that can understand spoken English which is the main goal to solve.One of the primary threats in evolving a question-answering model employing the SpokenSquad dataset is dealing with errors and noise in spoken language data. One option is using cutting-edge language models trained to overcome such difficulties, like BERT.

Natural Language Processing and Information Retrieval comprise Extractive Question Answering as a component. In Extractive Question Answering, a context is given so that the model may make assumptions about it and determine where the answer will be in the passage.Popular question-and-answer dataset SQuAD is employed in problems involving natural language analysis. Its purpose is to assess a model's capacity to respond to queries based on a particular passage of text. Researchers and developers can utilize two scripts, run qa.py and run tf squad.py, to interact with this dataset and change and train their models in accordance with the SQuAD task specifications.

**Description of model:**

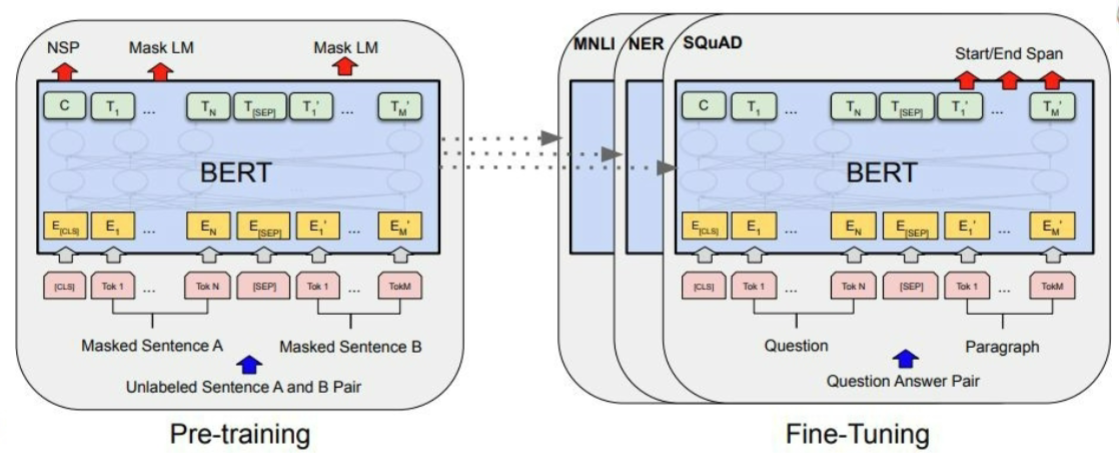
We have chosen the BERT (Bidirectional Encoder Representations from Transformers) model for this particular purpose.

Hugging Face is a well-known NLP library that offers models that have already been trained for different NLP tasks, which also involves question-answering. A pre-trained model is a neural network trained on a lot of data to carry out a specific task.The BERT is typically one of the models used as the primary model from Hugging Face for question-answering tasks.Masked language modeling (MLM) and next sentence prediction are used to train BERT, a potent natural language processing model, on a large corpus of text data (NSP). BERT may learn and develop its comprehension of the connections between words and sentences in natural language through this training process.

The BERT model is built from layers of feed-forward neural networks, positional embeddings, and self-attention.By paying attention to other words in the input sequence, the self-attention layers enable the model to capture the context of the terms. Feed-forward neural networks assist in converting the input into a high-dimensional space so that subsequent layers can easily handle it. The positions of each word in the input sequence are encoded using positional embeddings.

Hugging Face provides various pre-trained BERT-based models for question-answering, such as "Bert-large-uncased-whole-word-masking-finetuned-squad" and "bert-base-uncased," which have been fine-tuned using a large dataset like SQuAD to improve their accuracy in answering questions.

Using unlabeled input, the BERT pre-training model can generate highly effective bidirectional text representations. It leverages the left and proper contexts at all levels of the model architecture to create these representations. Adding just one output layer allows the pre-trained BERT model to be adapted and optimized for various NLP tasks, including language inference and question-answering. Empirical studies have shown that BERT is a straightforward yet compelling model for natural language processing tasks.



**Data set used:**

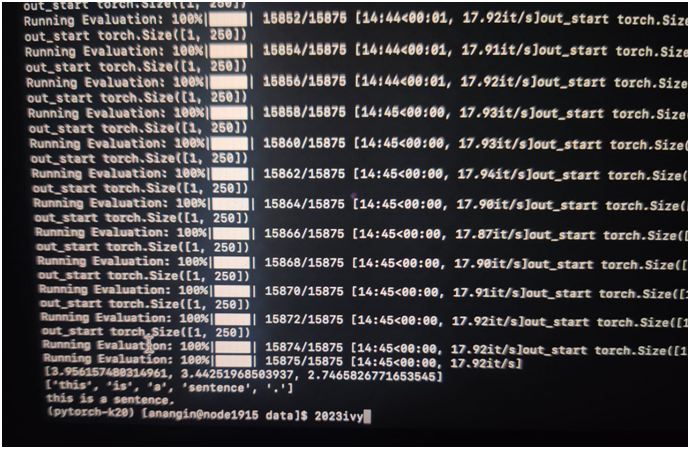
A well-liked benchmark dataset for assessing question-answering models is the Stanford Question and Answering Dataset (SQuAD). It consists of a sequence of questions and answers compared to a series of sections from Wikipedia entries. The aim is to create a model that can read the passage and respond to the prompt using the data it contains. Squad has gained substantial traction in natural language processing as a benchmark for assessing the effectiveness of question-answering algorithms.

**Outcomes:**

Word Error Rate (WER) is a statistic frequently used in speech recognition to quantify the difference between the predicted transcript and the reference transcript in terms of the number of errors. Hence it is not typically used to evaluate the question and answering systems.

When evaluating question-answering models, the F1 score is the most widely used metric, as it considers both precision and recall. This metric measures the degree of overlap between the predicted answer and the actual answer.

When it comes to the BERT model, evaluation datasets like the Stanford Question Answering Dataset often report the F1 score (SQuAD).



**Model improvement:**

When training a deep learning model, the learning rate is a crucial hyperparameter that determines the quantity of weights that should be adjusted during every iteration.When learning occurs at an abnormally rapid rate, the model may overshoot the optimal weights and fail to converge. Alternatively, when the learning rate is abnormally low , the model may connect deliberately or get stuck in a local minimum.

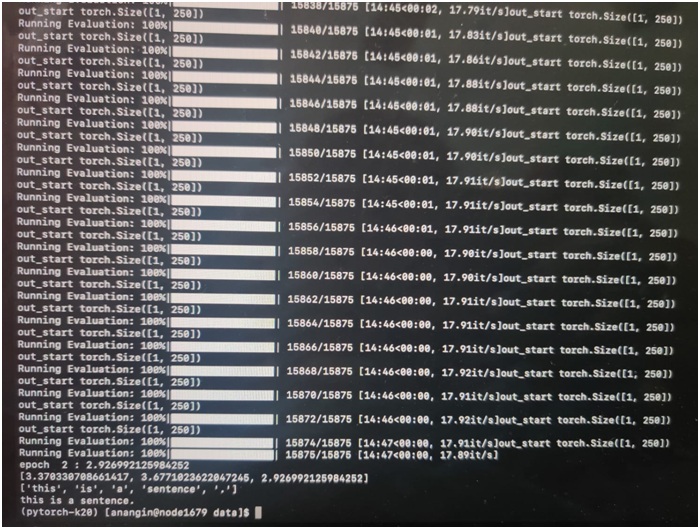
One type of learning rate schedule is the linear scheduler, which gradually decreases the learning rate with each epoch. This approach can help the model converge more effectively. To use a linear scheduler, specify the initial learning rate, the number of epochs, and the desired final learning rate. This will gradually reduce the learning rate throughout training.

Then, for each epoch, you can calculate the learning rate as follows:

lr = initial\_lr \* (1 - epoch / num\_epochs) + final\_lr \* (epoch / num\_epochs)

This equation calculates a linear interpolation between the initial and final learning rates based on the current epoch and the total number of epochs.In PyTorch, you can implement a linear scheduler using the LambdaLR class, which allows you to define a custom function to modify the learning rate at each epoch. The lr\_lambda argument is a lambda function that takes the current epoch as input and returns the corresponding learning rate.

Using a linear scheduler, you can fine-tune your model more effectively and achieve better performance on your task.



**Parameters and Analysis:**

Doc stride is a parameter used in natural language processing (NLP) activities, including text classification and question answering. It speaks about the separation between two successive textual or documental passages that are employed for analysis.

