NLP HW3

Ben Akhtar
906526538

November 26, 2023

1 ReadMe

The ReadMe can be found under the filename: README.md in the submission. Additionally, the following is the same replication information as in the ReadMe:

1.1 Instructions For Replication

1. Place the Jupyter Notebook file into google colab or an environment locally that has access to a GPU.
2. If you would like to run Roberta. Change cellblock 4 to roberta-large where it says bert-base-uncased.
3. There is additionally a codeblock that contains "huggingface_hub" that will require you to have a hugging face API token to login. This will allow you to save the model and be able to run it. This will need to be your own.
4. All other commands should run normally.

2 Working with the Data

For this homework, working with the data required a large amount of extra work. First, the dataset was loaded using the load dataset module in order to ensure the data was correct. From here, the dataset needed to manipulated to provide 4 different answers with the necessary context and question. The preprocessing function takes in the data and creates 4 new options from the 4 original answer choices. The new options all have the context and the question and then the answer is last. This is to ensure that the model has access to everything to predict the best possible answer.

From here, a custom data collator was used. This pads all of the inputs with the necessary SEP and PAD tokens from the tokenizer so that the model can properly read and accept our input. Here was not only tokenizing, but manipulating each answer beforehand to ensure the tokenizing would be properly done. Fed into the collator was the features and the tokenizer where the features included the inputs, the attention mask and the correct output label to ensure the model had everything it needed to make a prediction.

3 Building the Model

Most of the model building was followed using the tutorial found here: RocketKnight1.

The model itself is built off Bert, with some customizations. The first reason for using Bert is it will take far less time than Roberta while providing similar performance in our case. Additionally, the customization is so that we will be able to obtain our final outputs. These will then be compared to the labels so we can predict our accuracy on the task.

Once the model is built, the training loop is created, this was done using the Trainer library. This makes it easy to pass arguments and to predict the results compared to working with the raw PyTorch. 3 epochs were run to give a reasonable score and the context, question, and options were fed into the model after they had been properly encoded. From here, the prediction was made based on the question, context, and the choice of answers to give us the choice of answer between A-D or 1-4. This is then compared to the label and repeated.

4 Analysis

The reason Bert was chosen, and this was a good tutorial to follow, is that Roberta does better on question answering tasks than Roberta and especially Distilbert. Even though Roberta should do better, based on my experimental results, which yielded around 25% on Roberta vs the 55% on Bert.

I will discuss Roberta, as this is what should be the most ideal. Even though Roberta is quite large, it was almost necessary already to use Bert, so moving up to Roberta would make sense if given higher performance. Roberta has a dynamic masking strategy that created a more developed way to predict words since it had to use context to guess the words. This is perfect for something in the question answering task where the model has to guess the answer based on the context and question given. Since Roberta has this feature and Bert and Distilbert do not, it was deemed the best possible model to use before seeing results. After seeing results, Bert was used as it performed better.

On my Nvidia 3080, it took about 4 times the amount of time to train Roberta compared to Bert, which would be a factor in choosing Bert if capped for resources.

Other models that were tailored towards high-level textual analysis and especially guessing words based on context would have also likely done well for this task. They may not have been easily used however, and taken longer to run. Additionally, Bert and Roberta have a large amount of resources on the web for this task.

5 Evaluation

After training the model and the predict function was used on our model. Test was not done since it requires extra work and an account that I would not like to create. Test is assumed to be similar to validation. This achieved the following results:

Roberta-Large train: 0.25
Roberta-Large validation: 0.25
test: Assume similar to validation

Bert-Base-Uncased train: 0.9900641441345215
Bert-Base-Uncased validation: 0.575209379196167
test: Assume similar to validation

References

RocketKnight1. Fine-tuning a model on amultiple choice task. URL https://github.com/huggingface/notebooks/blob/main/examples/multiple_choice.ipynb.