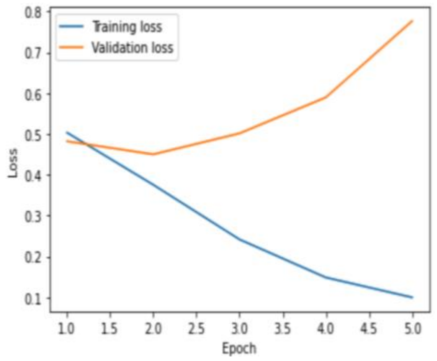
Project 2
Classification Approach.

0. **Description**
   In this section, we processed data and trained model in two ways: (1) In the first approach, we process the each option in the multiple choice questions as an independent input. The format is “fact + [SEP] + stem + [SEP] + one of the options + [SEP]”. The label for each input is whether the option is correct (1 or 0). For this approach we used a linear layer on top of BertModel with output size of 1. When calculating accuracy, we group 4 options together and count groups where all options are predicted correctly as correct. (2) In the second approach, we processed all options into one input in format “fact + [SEP] + stem + [SEP] + OptionA + [SEP] + OptionB+ [SEP] + OptionC+ [SEP] + OptionD”, and the label as a vector of size 4, which is the one hot encoding of correct option.
1. **Model Design**
   There is a defined model named ‘BertClassifier’. The model consists of two main components: a ‘BertModel’ layer and a’ linear classification’ layer.During training, the model is trained using a ‘BCEWithLogitsLoss’ loss function and ‘AdamW’ optimizer.
2. **Hyperparameters Setting:**
   ● Learning rate: 2e-5
   ● Batch size: 32
   ● Number of epochs: 5
   ● Max length of input text: 256
   ● Tokenizer: BertTokenizer (from 'bert-base-uncased')
3. **Performance:**
   Approach 1:
   ● Zero-shot accuracy: Validation accuracy: 0.00%, Test accuracy: 0.00%
   ● Fine-tuned accuracy: Validation accuracy: 37.80%, Test accuracy: 41.00%
   ● Training Time: ~40 mins



   Approach 2:
   ● Zero-shot accuracy: Validation accuracy: 27.40%, Test accuracy: 25.40%
   ● Fine-tuned accuracy: Validation accuracy: 23.20%, Test accuracy: 26.80%
   ● Training Time: ~40 mins
4. **Limitations and Solutions:**
   4.1 zero-shot accuracy is zero: For approach 1, before there is no fine-tuned model, Threshold for binary classification is set to 0.5, but the output probabilities for the

validation(test) set are all less than 0.5, which means the model is not confident enough to predict the input.
This is because the model has not been trained on the specific task of answering multiple-choice questions. The model input data is adjusted to include four options for a complete question, and the best option for each question is obtained through model training, and Using softmax to ensure that the sum of all option probabilities is 1 can change the problem that zeroshot is 0, which is approach 2. However, although approach 2 provides good results after fine-tuning in train data, the test and validation performance are poor. A possible explanation is that the input format are not in usual pattern as the text trained by Bert model. Also, the model slightly overfit, from the discrepancy between test and train performance.

4.2 Improve Performance

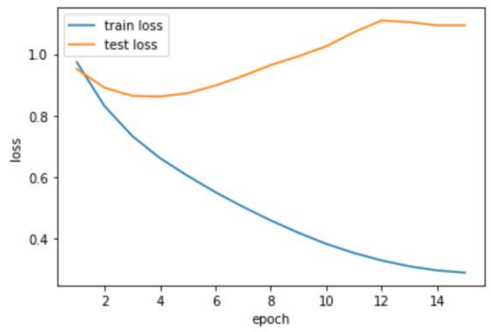
1. Increase model capacity: use a larger pre-trained model like 'bert-large-uncased'
2. Modify the hyperparameter: increase learning rate, batch size, epoch
3. Investigate the input data and consider modifying the way it is represent, e.g. changing the tokenization or adding more context to the input

Generative Results

Description

We process all data (training, validating, testing) with the same approach, which is constructing an input of "fact + [SEP] + stem + [SEP] + OptionA + [SEP] + OptionB+ [SEP] + OptionC+ [SEP] + OptionD"; then we added a space and the correct answer option (A/B/C/D) to the input as our labels. We utilized GPT2Tokenizer of "gpt2" checkpoint to tokenize our inputs and labels. The max_length is set to 128, which is found to be an appropriate length which does not truncate too many sequences. The tokenizer truncated and padded the sequences to max_length with "<pad>" special token as our padding token.
Then, we fine-tuned the GPT2LMHeadModel of "gpt2" checkpoint. In each training step, we passed in the training inputs and labels. In eval steps, the model generate one additional token and we compare the generated token with correct answer option (A/B/C/D) and calculated the accuracy.

1. Learning Curves:



2. Results

Zero-shot

The zero-shot accuracy for validation and test data are all 0.0. The reason for this zero accuracy is mainly because the model is not taught to generate a single option immediately after our input sequence. Instead, the zero-shot model keep generating meaningless tokens like “<pad>” or “[”, which cannot be used as multiple choice answers.

Fine-tuned

After training the data, the model is learned to generate an option after the input sequence. Our best model achieved a validation accuracy of 0.340 and a test accuracy of 0.354. The training time is approximately 5 minutes every epoch, which is 75 minutes in total.

3. Limitations

According to learning curves, the model overfits a bit in latter epochs. Therefore, we utilized learning-rate-decay technique to lower the lr in latter epochs to achieve better results.

4. Comparison with Classification Approach

From the perspective of test and validation performance, Generative approach perform better than Approach 2 in Classification Approach but slightly not as good as Approach 1. However, the training time, while all using Colab Tesla GPU, is much longer than Classification Approach. As a result, we believe using Classification Approach is a more efficient and suitable way to work on Multiple Choice Questions.