SI 630 HW5 Report

Dehao Zhang - Kaggle Username: dehaozhang
Xinhao Liao - Kaggle Username: Xinhao Liao

Part 1: Generating the start of a story

The script for this part is shown in *Final_p1.ipynb* and we run it on colab with GPU.

For the language model for language generation, we take advantage of the script *run_language_modeling.py*, and try to train an *openai-gpt* model.

After hyperparameter tuning, the model that performs the best (lowest perplexity) on the development set has the following key parameters:

1. Learning rate - 5e-5
2. Weight decay - 0.1
3. Block size - 512

All the unmentioned parameters are kept the same as default in languange_modeling.py.

Key evaluation metrics of the best model:

a. Loss on train.txt: 2.22238. Perplexity on train.txt: 9.2293
b. Perplexity on dev.txt: 7.7283
c. Perplexity on test.txt: 9.0620

Note: A key point for training the model is adding the argument '--line_by_line' in the script 'run_language_modeling.py'. By this argument, the perplexity on test.txt drops from around 30 to 9.0620. The reason could be that the provided text is actually independent across lines.

10 generated stories (repetition penalty = 1.2):

We use the *run_generation.py* for text generation. And we obtain 10 sentences with different input as follows.

a. My
My movie tells the story of an old captain, and his last days as a general of the indian army during world war i. he begins a long struggle for survival

b.The
The film opens with a limousine driver, jay, with his friend and potential mate, riding in it. jay agrees to a simple plan of the day and throws himself

c.One
One child has been able to learn a wonderful thing, but he can do no more than act like it does not matter. it is part of the cycle of agony

d.When (repetition penalty = 1.5)
When she comes out, aisha's life is changed forever. from the first day of school to cheerleading lessons, ajay is devastated. he believes that time

e.If
If his bet is a winner, the cowboy and the ranch hand then decide to find out what kind of rancher roy mcewan might be. after spending years

f.Our
Our princess starts out the cat and mouse game, that everyone on the planet is looking for. some people accidentally bump into. at this point, i guess some guy

g.First
First grown from the quadriplegic youth, alrel and his team of friends stop at the birth canal station, a nearby field called " the express lane "

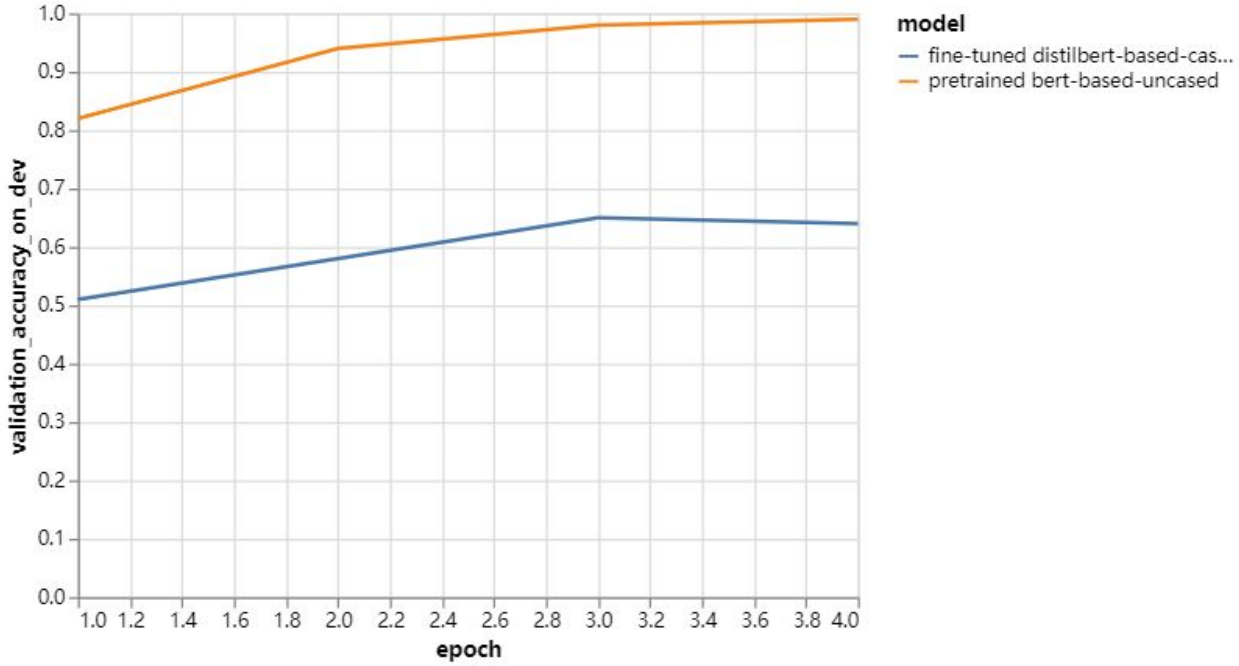
h.Natural
Natural french gardener mr. neville has a slip, and without the money he had been promised the previous year by colonel sieine, well, the pipe mr neville

i.We
We go out of the garden and to the house. we are seen taking up their bows, filling them up with water. they then shoot, and kill someone who

j.Because
Because of his luck, anthony delantan, a popular and intense writer, has taken to becoming an astronaut after the 70s. already, he has gained influence

Part 2: Training a classifier to recognize machine-generated text

Initially we try to fine-tune a "distilbert-base-cased" language model using *train.txt* in **Part 1**, and then train our classifier based on this model as instructed. But sadly, we finally found that the fine-tuned "distilbert-base-cased" (or "distilbert-base-uncased") model cannot have good performance, actually much worse than the pretrained "bert-base-uncased" model as shown below.



The line chart above clearly shows that the pretrained bert-based-uncased model has much better performance than our fine-tuned distilbert-based-cased model. The classifiers are both trained with learning rate = 2e-5 and adam_epsilon = 1e-8. When we tune these hyperparameters, both models show worse performances.

We also notice that unlike *train.txt* containing cased letters, *train.tsv* contains only uncased text. So we fine-tuned another *distilbert-based-uncased* language model and trained a classifier based on it. However, it shows performances similar to that of the *distilbert-based-cased* model. So being cased or not is not the key here.

When tested on *train.tsv*, the classifier trained from distilbert-cased model achieves the accuracy of 0.9972 and the one trained from the pretrained *bert-based-uncased* achieves the accuracy of 0.9996, which are both good performances.

When finally tested on *test.tsv* from Canvas and *test_text.tsv* from Kaggle, all the distilbert-cased models show bad performances stuck at around 0.55. In contrast, the pretrained *bert-based-uncased* model can achieve the test accuracy around 0.74.

We believe that the different performances mainly come from the model itself. Distilbert based models, as a light-version bert model, just cannot achieve as good performance as normal bert-based models. Unfortunately, we don't have enough memory to further fine-tune and pretrain the normal bert-based language model before training the classifier based on it.

Test on sentences generated in Part 1

The labels given by the classifier trained from the pretrained *bert-based-uncased* model is:

```
Human, Human, Machine, Human, Human, Human, Machine, Human, Machine,
Machine
```

which has an accuracy of only 0.4.

As for the classifier based on our fined-tuned and further trained distiltbert based model, it gives the labels as follows

```
Machine, Machine, Human, Human, Human, Human, Human, Human, Human, Human
```

which has an accuracy of only 0.2.

Clearly, our language generation somehow successfully cheats our text classifier.

The script for training and testing of the classifier based on our fine-tuned and further train distiltbert based model is given in *Final_p2_distilbert.ipynb*.

And the script for training and testing of the classifier based on the pretrained *bert-based-uncased* model is given in *Final_p2_bert.ipynb*.