Assignment #10000

In this assignment, I will compare recurrent and transformers based language models on IMDB dataset. The starter code is provided by Ankur Mali: <https://github.com/AnkurMali/IST597_Spring_2022/tree/main/assignment10000>

Based on the start code, I tried different optimizers and learning rates and recorded the accuracy and time. In the LSTM, I split 20% of the training set as a validation set. The fixed parameters are: batch size = 512, cell size = 64, dense size = 128, epoch = 10. Below is the summary.

LSTM

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Optimizer | Learning rate | Train accuracy | Time in second | Test acc |
| Adam | 0.0001 | 0.9777666 | 874 | 0.8670857 |
| Adam | 0.001 | 0.9820518 | 862 | 0.8716293 |
| Adagrad | 0.0001 | 0.62377405 | 973 | 0.6149825 |
| Adagrad | 0.01 | 0.98043257 | 986 | 0.8660032 |
| GradientDescent | 0.0001 | 0.51544267 | 835 | 0.52040726 |
| GradientDescent | 0.01 | 0.52512318 | 823 | 0.51986137 |
| RMSProp | 0.0001 | 0.9356137 | 861 | 0.8774559 |

We can see that Adam and RMSProp both give relatively good accuracy. Adagrad has poor accuracy when learning rate is 0.0001. When I changed the lr to 0.01, the performance is much better. It shows that Adagrad needs a small learning rate to start with. For Gradient Descent, I tried different lr from 1e-4 to 1e-2, but the accuracy is always very low. I was not very sure about what went wrong. From speed perspective, Gradient Descent is the fastest. Adagrad is the slowest. Adam and RMSProp have similar speed. To conclude, both Adam and RMSProp are very good overall. They have relatively high accuracy and fast speed. Although Gradient Descent is the fastest, its accuracy is very poor.

Next, I used a small bert: bert\_en\_uncased\_L-4\_H-512\_A-8. The fixed parameters are dropout = 0.1, epoch = 10. I tried two types of optimizers and different initial learning rates. Then a fine-tune is run. Below is the summary.

Bert

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| optimizer | Init\_lr | Train acc | Time | Test acc |
| adamw | 3e-5 | 0.9406 | 3142 | 0.85648 |
| adamw | 2e-4 | 0.9988 | 3113 | 0.85129 |
| adamw | 2e-5 | 0.9998 | 3241 | 0.84656 |
| Lamb | 2e-5 | 0.8644 | 3215 | 0.84068 |
| Lamb | 2e-4 | 0.9989 | 3267 | 0.84988 |

We can see that all of them have similar and relatively good performance. The speed is also similar. I think it is because both adamw and lamb are Adam optimizers. Therefore the results tend to be similar. I only issue I found is that after the first model, later models have very high train accuracy. See rows 1 and 2, 3. This looks like an overfitting and I do not know what is the reason. Below I also have training vs validating plots. The left plot corresponds to row 1 in the table. The right plot corresponds to row 3 in the table. We can see from the right plot that there might exist overfitting.

Chart, line chart

Description automatically generatedGraphical user interface, application

Description automatically generated

To compare bert and lstm, we can conclude that lstm has better performance on both speed and accuracy. The fine-tune on bert takes a longer time but produces no better results than lstm.