Kevin Hsieh

2078611

COSC4368

Dr. Eick

Problem Set 2 Task 4:

Before we can use our data in a model, the data needs to be processed into an acceptable format for the model. A model does not understand raw text, images or audio. These inputs need to be converted into numbers and assembled into tensors. In this task, we preprocess the text with a tokenizer called the BertTokenizer of which we import from the transformers package.

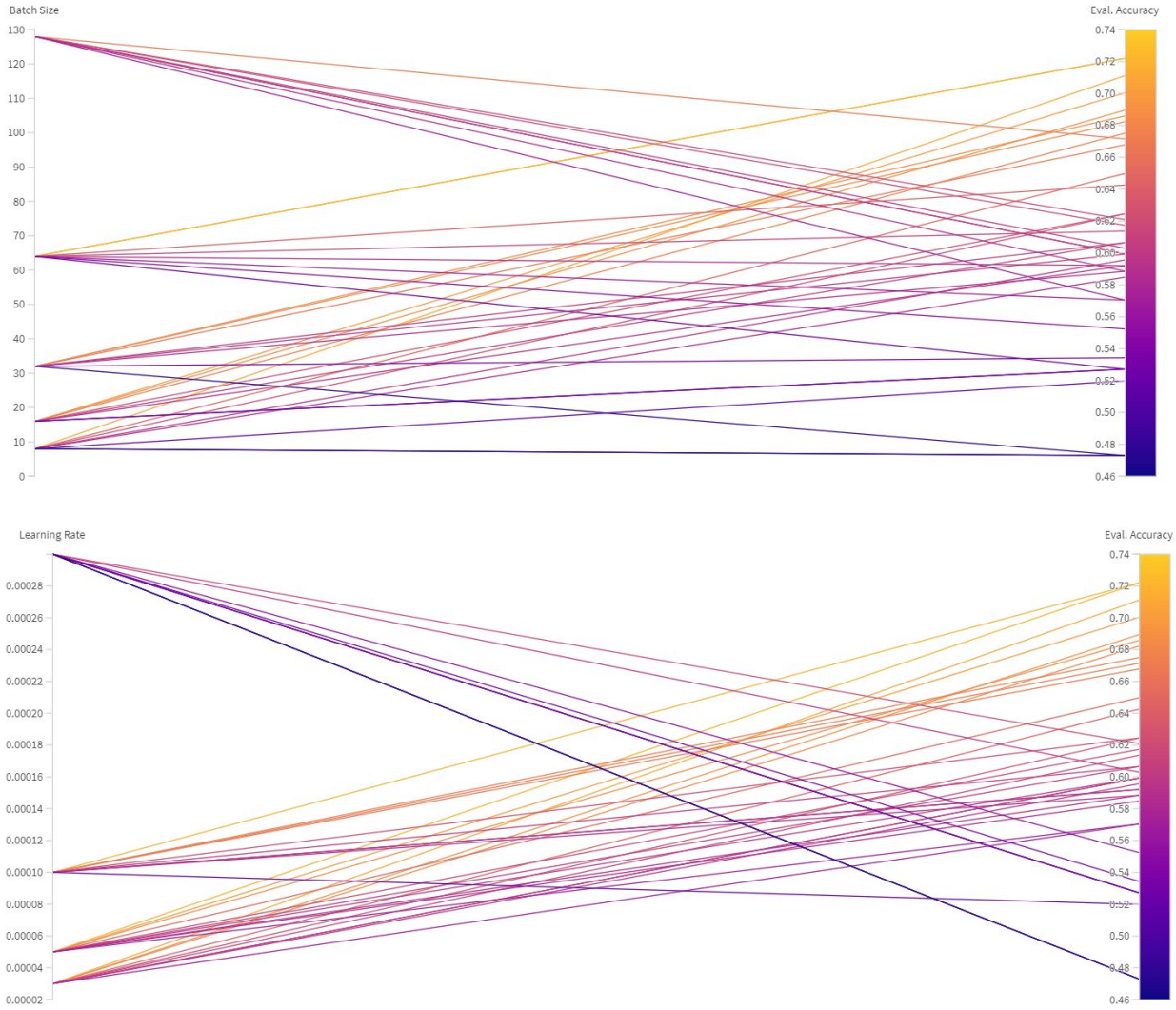
BERT relies on a Transformer (the attention mechanism that learns contextual relationships between words in a text). A basic Transformer consists of an encoder to read the text input and a decoder to produce a prediction for the task. Since BERT's goal is to generate a language representation model, it only needs the encoder part. The input to the encoder for BERT is a sequence of tokens, which are first converted into vectors and then processed in the neural network. But before processing can start, BERT needs the input to be massaged and decorated with some extra metadata:

1. Token embeddings: A [CLS] token is added to the input word tokens at the beginning of the first sentence and a [SEP] token is inserted at the end of each sentence.

2. Segment embeddings: A marker indicating Sentence A or Sentence B is added to each token. This allows the encoder to distinguish between sentences.

3. Positional embeddings: A positional embedding is added to each token to indicate its position in the sentence.

For each of the BERT models, bert_uncased_L-2_H-128_A-2, bert_uncased_L-4_H-256_A-4, bert_uncased_L-4_H-512_A-8, and bert_uncased_L-8_H-512_A-8, I used a batch_size of 50, epoch of 1, and learning rate of 3e-5. I chose this hyperparameter because relative to batch size, learning rate has a much higher impact on model performance. So if we're choosing to search over potential learning rates and potential batch sizes, it's probably wiser to search spend more time tuning the learning rate. The learning rate has a very high negative correlation (-0.540) with model accuracy. Therefore, in our search space, lower learning rates are better.



This confirms what the Parameter Importance plot already told us: low learning rates are good, batch size doesn't matter very much. We'd need more evidence to confirm, but I'd say that a smaller batch size is preferable in this case, too.

And here's our final results:

BERT Model Learning_rate: 3e-5 Epoch=1	Train Accuracy	Test Accuracy	Precision	Recall	F1-Score
BERT-Tiny (L-2_H-128_A-2)	0.8138	0.80496	0.806486	0.80496	0.804717
BERT-Mini (L-4_H-256_A-4)	0.8382	0.838122	0.838122	0.83812	0.838120
BERT-Small (L-4_H-512_A-4)	0.8524	0.8529	0.854810	0.85288	0.852680
BERT-Medium (L-8_H-512_A-8)	0.8602	0.8549	0.860484	0.85492	0.854358

In this task, we tokenized our review with our pre-trained BERT tokenizer. We then fed those tokenized sequences to our model and ran a final softmax layer to get the predictions. We then used the argmax function to determine whether our sentiment prediction for the review is positive or negative. Finally, we print out the results with a sklearn metrics call to classification_report and accuracy_score. We have successfully built a transformers network with 4 pre-trained BERT model and achieved ~86% accuracy on the sentiment analysis of the movie sentiment reviews dataset.