- **Specifies which option they chose (+0.5)** I decided to use the bert-base-uncased model, and fine-tune it for binary sentiment classification using the sst2 dataset.

  Dataset: https://huggingface.co/datasets/sst2/viewer/default/train

  Model: https://huggingface.co/bert-base-uncased

- **Description of the task and models (+1)** Binary sentiment classification is a natural language processing (NLP) task that involves analyzing and categorizing text data into one of two predefined categories: positive or negative sentiment. The primary objective of this task is to determine the emotional tone or sentiment expressed within a piece of text, such as a sentence, review, or document, and classify it as either positive (indicating a positive sentiment or emotion) or negative (indicating a negative sentiment or emotion).

  "BERT (Bidirectional Encoder Representations from Transformers) based uncased" refers to a variant of the BERT model, which is a powerful pre-trained neural network architecture for natural language processing (NLP) tasks. "Uncased" means that the model is trained on text data without distinguishing between uppercase and lowercase letters.

  The SST-2 dataset, short for Stanford Sentiment Treebank 2, is a collection of sentences and phrases from movie reviews. Each example in the dataset is labeled as either "positive" or "negative," indicating the sentiment conveyed by the text, making it useful for sentiment analysis tasks.

  The split of the sst2 dataset is very odd to me: 67k lines of text are reserved for training, with only 872 for validation, and finally 1821 are testing data. It seems lopsided, and strange that the testing data is unlabeled. So I used the validation data as my testing data.

  Additionally, the lines of text can contain strange typos, including randomly placed punctuation and spaces, like in row 82 of the training data: [START], like life , is n't much fun without the highs and lows [END]

  The concern I have is with the tokenizer seeing those spaces and making incomplete words, or having misspelled words fail to match anything in the model's vocabulary. This may cause inaccuracy.

  The way this could be rectified is by going through the dataset and fixing all the grammar; a tedious and expensive task. Additionally, grammatical errors exist in real data, so it may be good to train the model to handle such errors.

- **Includes appropriate references (+1)** See Works Cited
- **Mentions something about the hardware they used (0.5)** I am running this code on my own PC. I have a NVIDIA GeForce GTX 1660 graphics card, and 32 GB of RAM.
- **Explains how they checked if their model was trained correctly using learning curve graphs or other appropriate information (+2)**

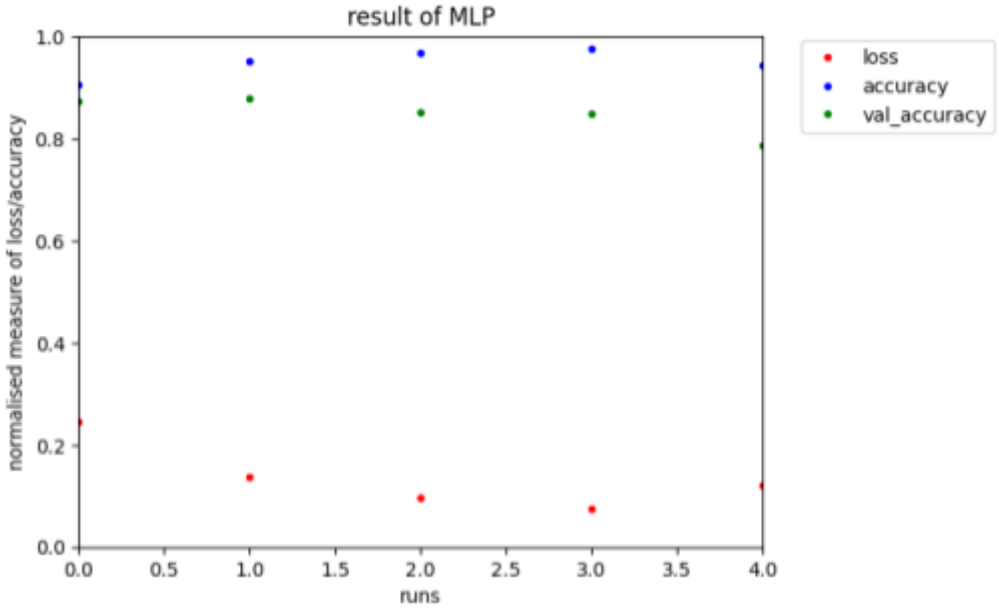

Figure 1: chart of metrics over 5 epochs

Using this graph of loss and accuracy, I checked to make sure that my model was improving. The most important metric was the final validation accuracy, or the accuracy on the testing data. I then used the finished model to classify some new data to double check that it was working.

It is clear that the model began overfitting from epoch 2, resulting in worse generalization

- **Specifies evaluation metrics used in experiment (+1)** Accuracy was the primary metric I used. It is a count of all the correct classifications the model made, divided by the total. In other words, when tested, the portion of correct classification.

  The other metric is the loss. The standard loss function for binary classification is binary cross-entropy loss, quantifying the dissimilarity between the predicted probabilities and the actual binary labels (0 or 1). The loss is used to update the parameters of the model, by calculating the partial derivative with respect to those parameters to see which changes would result in a higher score. However this metric is less understandable to humans.

$$-\frac{1}{N}\sum_{i=1}^{N} y_i \cdot log(p(y_i)) + (1 - y_i) \cdot log(1 - p(y_i))$$

Figure 2: Equation for Binary Cross-Entropy

- **Discusses test set performance and comparison with score reported in original paper or leaderboard. Includes justification if it differs from the reported scores. (+2)**

  I studied a paper called Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer, from the top of the sst2 sentiment classification leaderboard. This team achieved an impressive accuracy rate of 97%. Even with the same limitations mentioned above, this score far surpassed the performance of my own model.

  I can clearly see that my model began overfitting at the second epoch, which has contributed to low scores. Even before my model overfit, its peak metric score was only 87.8% accuracy on testing data. This implies something fundamentally different between my approach and that of my approach and the authors. Upon closer examination, I learned that the authors of the paper conducted a comprehensive investigation into various aspects, including pre-training objectives, architectural choices, unlabeled datasets, and transfer techniques.

- **Includes training and inference time (+0.5)** Training time was 207 minutes, inference time was about 26 seconds.

- **Includes hyperparameters used in experiment (+1)** I used 5-epochs, which took a long time on the massive 67 thousand dataset I used. The low amount of epochs was both to reduce training time, and to avoid overfitting on that massive dataset since by the 1st epoch the metrics would be high.

- **Hypothesis of model performance and/or some kind of discussion about what they found in their incorrectly labeled samples (+1)**

  In my analysis of the mistakes made on the validation data, I found 41 false positives and 144 false negatives. This suggests that the model is far too conservative. This could be a result of overfitting during the training.

  The model incorrectly labeled a sentence that criticizes or negatively reviews a film as positive. This suggests that the model might have difficulty understanding sarcasm or negative sentiment.

- **Minimum of ten incorrectly predicted test samples with their ground-truth labels (+1)**

  False Positives:

  the title not only describes its main characters , but the lazy people behind the camera as well .

  for all its impressive craftsmanship , and despite an overbearing series of third-act crescendos , lily chou-chou never really builds up a head of emotional steam .

  this is a train wreck of an action film -- a stupefying attempt by the filmmakers to force-feed james bond into the mindless xxx mold and throw 40 years of cinematic history down the toilet in favor of bright flashes and loud bangs .

  you wo n't like roger , but you will quickly recognize him .

  his last movie was poetically romantic and full of indelible images , but his latest has nothing going for it .

  there is n't nearly enough fun here , despite the presence of some appealing ingredients .

  it seems to me the film is about the art of ripping people off without ever letting them consciously know you have done so

  the tale of tok (andy lau) , a sleek sociopath on the trail of o (takashi sorimachi) , the most legendary of asian hitmen , is too scattershot to take hold .

  the character of zigzag is not sufficiently developed to support a film constructed around him .

  the talented and clever robert rodriguez perhaps put a little too much heart into his first film and did n't reserve enough for his second .

  False Negatives:

  we root for (clara and paul) , even like them , though perhaps it 's an emotion closer to pity .

```
seldom has a movie so closely matched the spirit of a man and
his work .

mr. tsai is a very original artist in his medium , and what
time is it there ?

this is a story of two misfits who do n't stand a chance alone
, but together they are magnificent .

if you 're hard up for raunchy college humor , this is your
ticket right here .

good old-fashioned slash-and-hack is back !

one of creepiest , scariest movies to come along in a long ,
long time , easily rivaling blair witch or the others .

there is nothing outstanding about this film , but it is good
enough and will likely be appreciated most by sailors and folks
who know their way around a submarine .

though it 's become almost redundant to say so , major kudos go
to leigh for actually casting people who look working-class .

jaglom ... put ( s ) the audience in the privileged position of
eavesdropping on his characters
```

- **Discusses potential modeling or representation ideas to improve the errors (+1)**

  Here are some ideas for ways to reduce overfitting in my model:

  Dropout: Introduce dropout layers to reduce overfitting by randomly deactivating some neurons during training.

  Another idea is L1 and L2 Regularization: Add penalty terms to the loss function to discourage overly complex model weights.

  Early Stopping my monitoring the validation loss during training and stop training when it starts to increase. This prevents the model from continuing to fit noise in the data.

  Cross-Validation: Use k-fold cross-validation to assess the model's performance on different subsets of the data. This helps you get a better estimate of its generalization capabilities.

- **Lists contributions from each member (+0.5)** I did the project solo

Works Cited:

I used the following resources to complete this homework:

- Shirley's Presentation on September 19: https://docs.google.com/presentation/d/19AVdYw2ezOprQuBBumICEbfNpvJ-pVVURiOb7aQDjdo/edit#slide=id.g27c35a91bda_0_0
-
- Shirley's jupyter notebook: https://colab.research.google.com/drive/1sMCEi6zXP8AQLoCrXf-QmiwoXB5v57wy
-
- PyTorch programming tutorial from September 14: https://docs.google.com/presentation/d/1DNWnMJYswVhzo7hmI9mZLNhjPc8rNXsERM1bvFSE6Fo/edit#slide=id.g27d93a5fe38_0_92
-
- PyTorch programming tutorial jupyter notebook: https://colab.research.google.com/drive/1Mely63oAnovTKoW82w1NuVJvs9HH5dqd?authuser=1
-
- The homework assignment's custom trainer: https://dykang.github.io/classes/csci5541/F23/hw/CSCI_5541_F23_HW1.pdf
-
- Hugging Face Tutorial: https://huggingface.co/docs/transformers/training
-
- Original Paper: https://arxiv.org/pdf/1910.10683v4.pdf

Chat GPT Prompts:

- write me a brief description of the NLP task binary sentiment classification
-
- How can I get the model's predictions from the input data during the forward pass when fine-tuning a pre-trained model?
-
- For loss calculation using CrossEntropyLoss, what's the correct way to compute the loss between the model's predictions (logits) and the labels?
-

- Could you provide guidance on implementing an effective evaluation loop to assess the model's performance on the evaluation dataset, including both loss and accuracy calculations?
-
- give an explanation of how binary cross-entropy is calculated, with an equation