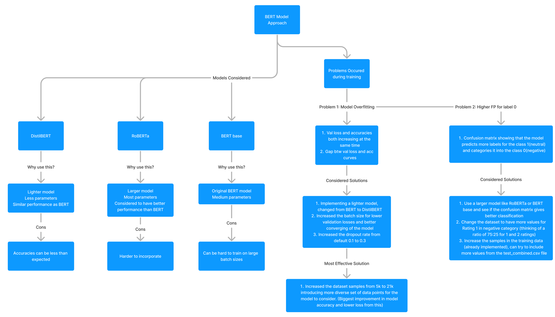
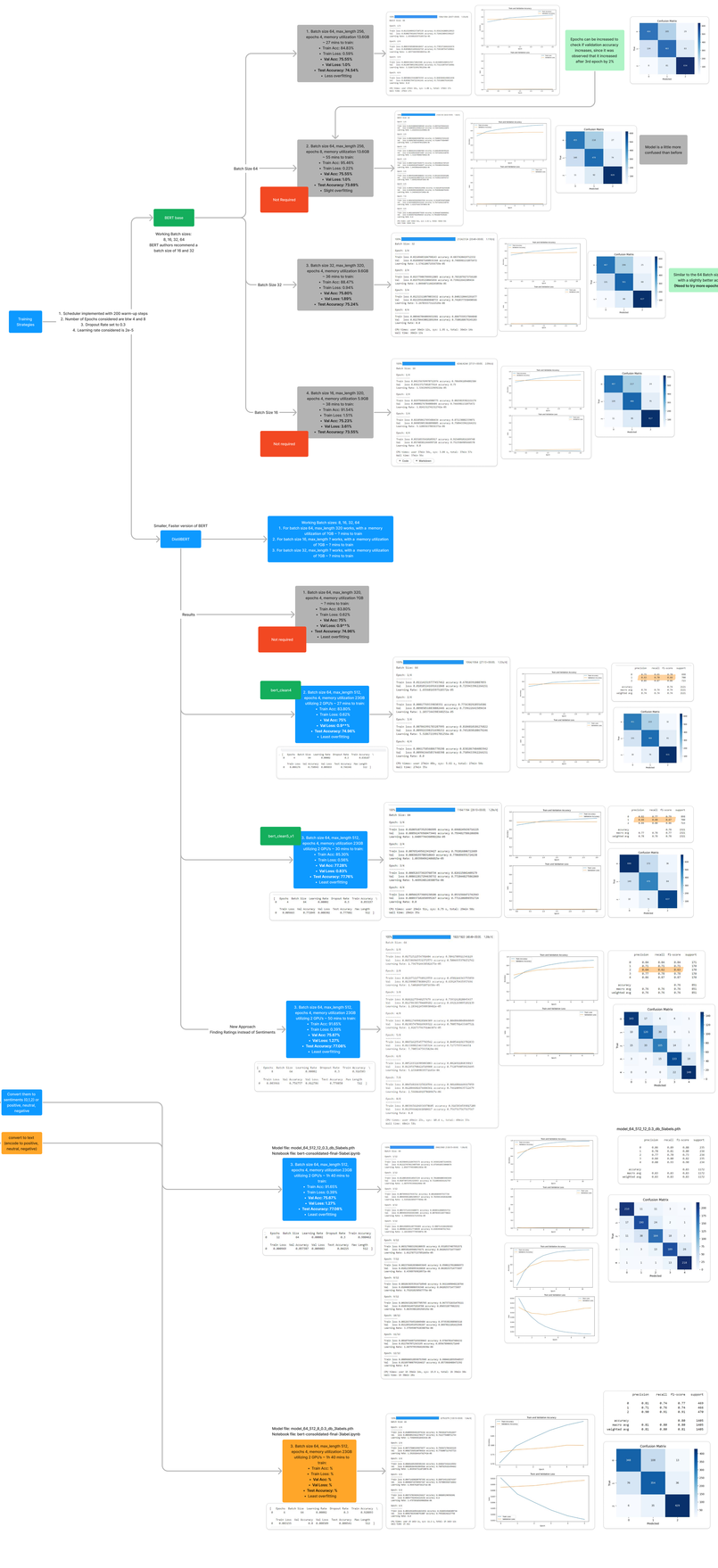

BERT Model Approach
Models Considered
Problems Occured during training
DistilBERT
RoBERTa
BERT base
Why use this?
Lighter model
Less parameters
Similar performance as BERT
Cons
Accuracies can be less than expected
Larger model
Most parameters
Considered to have better performance than BERT
Harder to incorporate
Original BERT model
Medium parameters
Can be hard to train on large batch sizes
Problem 1: Model Overfitting
Problem 2: Higher FP for label 0
1. Val loss and accuracies both increasing at the same time
2. Gap b/w val loss and acc curves
Considered Solutions
1. Implementing a lighter model, changed from BERT to DistilBERT
2. Increased the batch size for lower validation losses and better converging of the model
3. Increased the dropout rate from default 0.1 to 0.3
Most Effective Solution
1. Increased the dataset samples from 5k to 29k introducing more diverse set of data points for the model to consider. Biggest improvement in model accuracy and lower loss from this
1. Confusion matrix showing that the model predicts more labels for the class 1(neutral) and categories 0 into the class 0(negative)
1. Use a larger model like RoBERTa or BERT base and see if the confusion matrix gives better classification
2. Change the dataset to have more values for Rating 3 to negative category (thinking of a ratio of 75:25 for 1 and 2 ratings)
3. Increase the samples in the training data (already implemented), can try to include more values from the test_combined.csv file


Training Strategies
1. Scheduler implemented with 200 warm-up steps
2. Number of Epochs considered are btw 4 and 8
3. Dropout Rate set to 0.3
4. Learning rate considered is 2e-5
BERT base
Working Batch sizes: 8, 16, 32, 64
BERT authors recommend a batch size of 16 and 32
Batch Size 64
1. Batch size 64, max_length 256, epochs 4, memory utilization 13.6GB
• 27 mins to train
• Train Acc: 84.81%
• Train Loss: 0.595
• Val Acc: 74.94%
• Val Loss: 1.05
• Test Accuracy: 74.54%
• Less overfitting
Epochs can be increased to check if validation accuracy increases, since it was observed that it increased after 3rd epoch by 2%
2. Batch size 64, max_length 256, epochs 4, memory utilization 13.6GB
• 55 mins to train
• Train Acc: 95.46%
• Train Loss: 0.225
• Val Acc: 79.55%
• Val Loss: 1.05
• Test Accuracy: 73.88%
• Slight overfitting
Not Required
Model is a little more confused than before
Batch Size 32
3. Batch size 32, max_length 320, epochs 4, memory utilization 9.0GB
• 38 mins to train
• Train Acc: 88.47%
• Train Loss: 0.345
• Val Acc: 79.80%
• Val Loss: 1.09%
• Test Accuracy: 79.24%
Similar to the 64 Batch size model, with a slightly better accuracy (Need to try more epochs for this)
Batch Size 16
4. Batch size 16, max_length 320, epochs 4, memory utilization 5.9GB
• 38 mins to train
• Train Acc: 93.71%
• Train Loss: 1.575
• Val Acc: 75.23%
• Val Loss: 3.85
• Test Accuracy: 73.55%
Not required
Smaller, Faster version of BERT
DistilBERT
Working Batch sizes: 8, 16, 32, 64
1. For batch size 8, max_length 320 works with a memory utilization of 1GB ~ 7 mins to train
2. For batch size 16, max_length 5 works, with a memory utilization of 1GB ~ 7 mins to train
3. For batch size 32, max_length 5 works, with a memory utilization of 1GB ~ 7 mins to train
Results
1. Batch size 64, max_length 320, epochs 4, memory utilization 1GB
• 7 mins to train
• Train Acc: 89.67%
• Train Loss: 0.627
• Val Acc: 75%
• Val Loss: 0.97%
• Test Accuracy: 74.06%
• Least overfitting
Not required
bert_clean4
2. Batch size 64, max_length 512, utilizing 3 GPUs ~ 27 mins to train
• Train Acc: 89.67%
• Train Loss: 0.627
• Val Acc: 75%
• Val Loss: 0.97%
• Test Accuracy: 74.06%
• Least overfitting
bert_clean4_v1
3. Batch size 64, max_length 512, epochs 4, memory utilization 23GB utilizing 3 GPUs ~ 55 mins to train
• Train Acc: 89.30%
• Train Loss: 0.366
• Val Acc: 78.85%
• Val Loss: 0.63
• Test Accuracy: 77.76%
• Least overfitting
New Approach
Finding Ratings instead of Sentiments
3. Batch size 64, max_length 512, epochs 4, memory utilization 23GB utilizing 3 GPUs ~ 50 mins to train
• Train Acc: 91.65%
• Train Loss: 0.38%
• Val Acc: 75.67%
• Val Loss: 1.27%
• Test Accuracy: 77.08%
• Least overfitting
Train the model on predicting ratings
Predict Ratings 1 to 5
Convert them to sentiments 0,1,2 or positive, neutral, negative
Train the model on predicting sentiments (0,1,2)
Predict Sentiment (0,1,2)
convert to text (encode to positive, neutral, negative)
Model file: model_64_512_12_0_3_db_Stabels.pth
Notebook file: bert-consolidated-final-Stabel.ipynb
4. Batch size 64, max_length 512, epochs 4, memory utilization 23GB utilizing 3 GPUs ~ 1h 40 mins to train
• Train Acc: 91.65%
• Train Loss: 0.38%
• Val Acc: 75.67%
• Val Loss: 1.27%
• Test Accuracy: 77.08%
• Least overfitting
model_64_512_12_0_3_db_Stabels.pth
model_128_320_8_0_3_db_Stabels.pth
Model file: model_64_512_8_0_3_db_Stabels.pth
Notebook file: bert-consolidated-final-Stabel.ipynb
5. Batch size 64, max_length 512, epochs 4, memory utilization 23GB utilizing 3 GPUs ~ 52 mins to train
• Train Acc: %
• Train Loss: %
• Val Acc: %
• Val Loss: %
• Test Accuracy: %
• Least overfitting
3. Batch size 64, max_length 512, epochs 4, memory utilization 23GB utilizing 3 GPUs ~ 52 mins to train
• Train Acc: 84%
• Train Loss: 0.25%
• Val Acc: 79.67%
• Val Loss: 0.54%
• Test Accuracy: 81.96%
• BEST MODEL**