Assignment 1
NLP and Representation Learning (DS-GA 1011)

Script on GitHub

Name: Mihir Ujjwal Rana NetID: mur214

Bag of N-Grams Document Classification

All models were trained for 20 epochs, and training loss was stored 10 times in each epoch, while validation accuracy was computed once in each epoch.

Parameter grid used for cross validation:

BATCH_SIZEs = [32, 64], **Best: 64**
LRs = [1e-2, 1e-3], **Best: 1e-3**
EMB_DIMs = [100, 512, 1024], **Best: 512**
VOCAB_SIZEs = [10000, 20000, 50000], **Best: 10000**
OPTIMIZERS = [SGD, ADAM], **Best: Adam**
MAX_SENTENCE_LENGTHs = [200, 500], **Best: 200**
NGRAMs = [1,2,3,4], **Best: 4**

Best Validation Accuracy: 87.92%
Best Test Accuracy: 86.14%

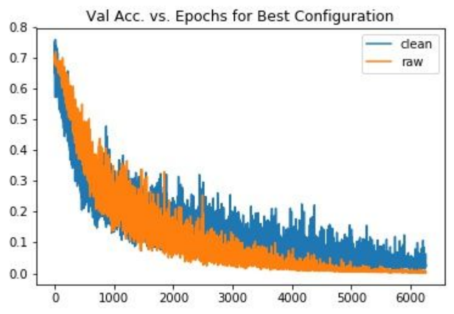
Note: All graphs are on the best configuration obtained listed above (except for the varying parameter), and on cleaned data.

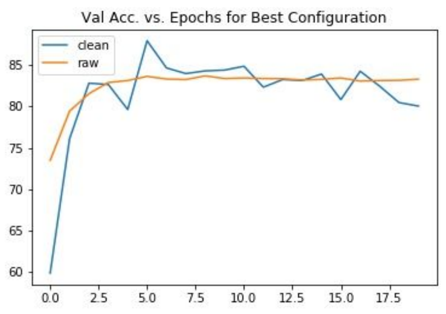
1. Tokenization Schemes

For tokenization schemes, two main differences were analyzed, one in which no preprocessing was done, and another in which the following things were removed:

- Stopwords (except for negating words, as they may account for sentiment), punctuations, HTML tags, accented characters (normalization), brackets, etc.
- Lowercasing
- Lemmatizing

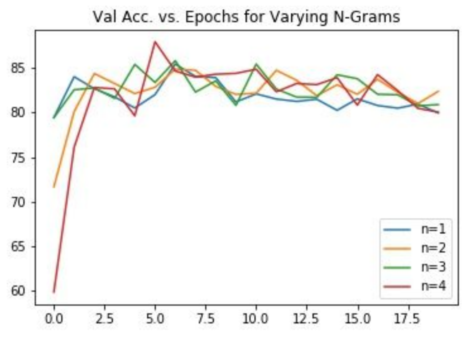
As expected, the latter cleaning in which cleaning was performed gave better validation accuracy, since we're removing characters/words that don't provide any useful information to the model.

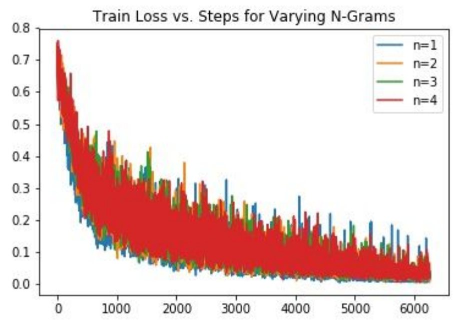




2. Vary ‘n’ in n-grams

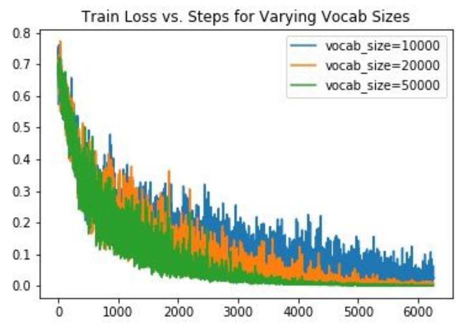
As can be seen in the ablation study below, on increasing the ngram range, the max accuracy increases (very slightly):

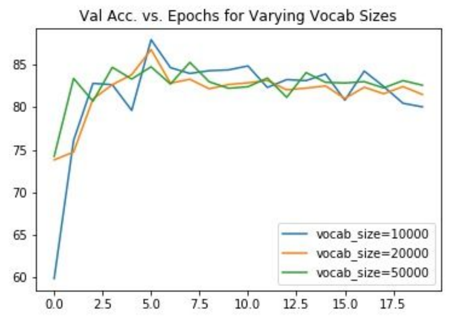




This may be due to the fact that we’re accounting for more structure in the data, and picking up common phrases (not just words). Higher n-grams also account for ordering to tokens to an extent.

3. Vary vocabulary size

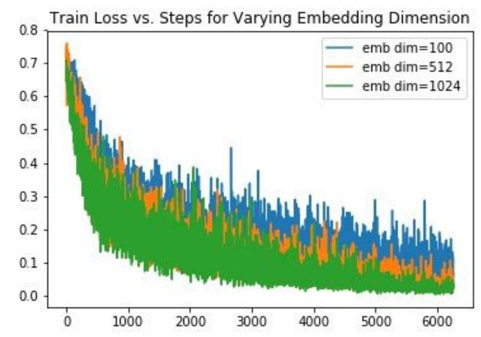


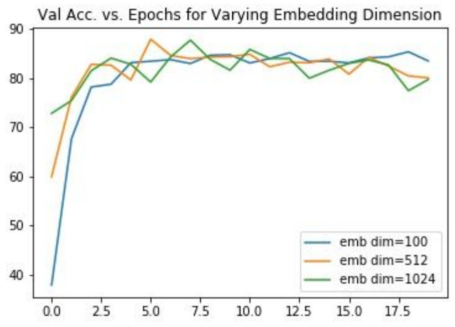


Vocab size = 10000 with upto 4-grams gives best results, although accuracies are fairly similar for all configurations. In this case, choosing the simplest model (e.g. 10000) could provide a substantial gain in speed and memory footprint.

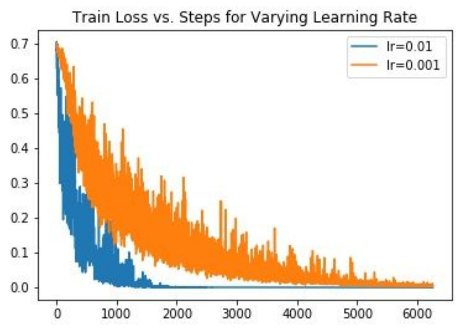
4. Vary embedding size

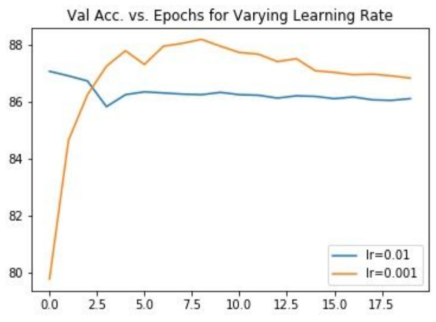
An embedding dimension of 512 yields best results. This might be since we’re allowing for a higher dimensional representation in the embedding space compared to 100, while a much larger dimension, like 1024, could also cause overfitting because it will be too many parameters for a relatively small data set (although in this case it doesn't seem to be overfitting for 20 epochs).





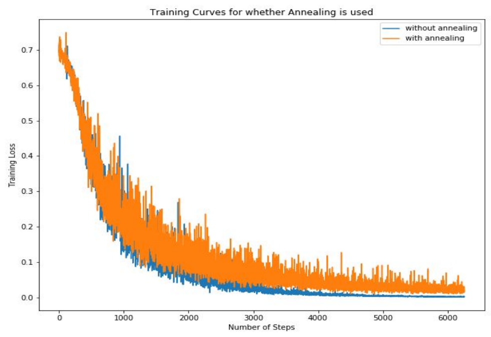
5. Vary Learning Rate

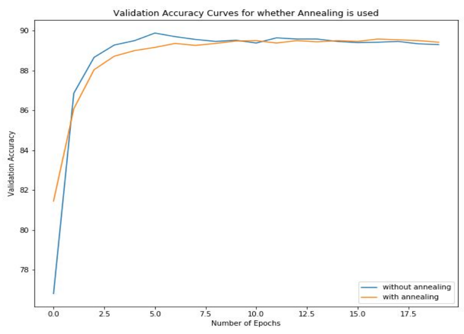




Although the training loss converges much faster in the higher learning rate of 0.01, it doesn't as good results on the validation set. Learning rate = 0.001 gives significantly better results, which is as expected, as 0.01 might be a little too big.

6. Annealing of Learning Rate

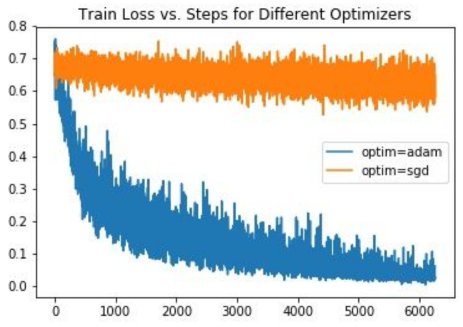


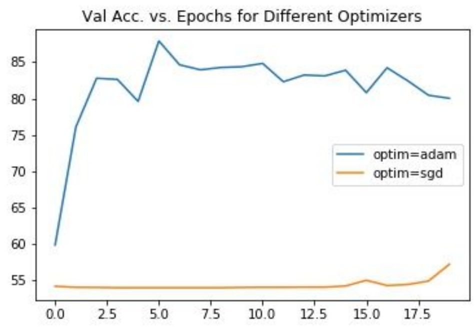


Linear annealing doesn't seem to help a lot in this case. However, a couple things to note:

- For my experiments, I used step_size = 1 and gamma = 0.9, but this is likely to have a significant impact
- Annealing helps more when more epochs are used, and it's likely that 20 weren't enough to see a gain

7. Vary Optimizer





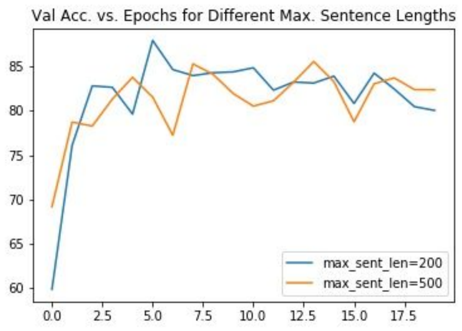
SGD performs very poorly (there is no learning in some cases, as seen above), while Adam gives much better results for both losses and accuracies. This is expected, since Adam usually always performs much better.

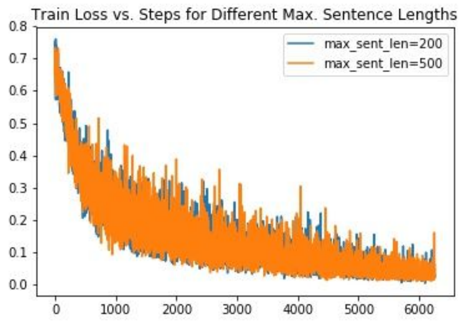
8. Correct/Incorrect Examples

See all examples here.

In all 3 misclassified examples, one could argue that the model fails to classify correctly because there is a reasonably big mix of both positive and negative sentiment oriented words, i.e., the language is very ambiguous, which means it is easy for the model to get confused.

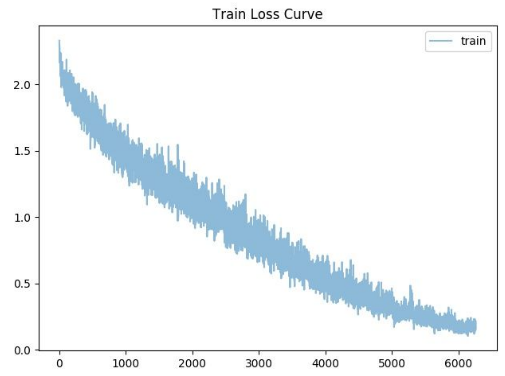
9. Vary Maximum Sentence Length

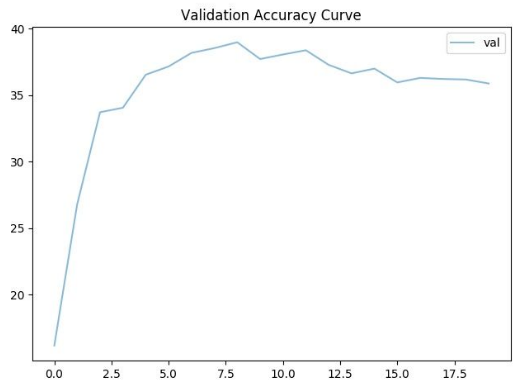




Maximum sentence length of 200 gives better results, although the two are fairly similar. The kind of tokens that finally make it into the feature space depend on the ngram range and the maximum sentence length, and this may be a cause of that.

10. Ratings Prediction





The validation accuracy obtained (although I didn't perform a comprehensive parameter search) is **38.4%**. Clearly, a simple BoW model is not powerful enough to predict the ratings, which is also aided by the fact that it has to now predict 10 classes instead of 2.