NLP-Assignment-3
Divita Phadakale

1. Run a few experiments with different embedding dimensions, batch sizes, and number of epochs run.

Solution:
After running hyperparameter loop for different values of embedding dimensions and batch sizes this is the F1 scores when evaluated against val data.

```
print(df_results)
```

```
    Embedding Dim  Batch Size  Num Epochs    Embedding Type  F1 Score
0              50          10          10    Own Embeddings  0.400000
1              50          10          10  GloVe Embeddings  1.000000
2              50          20          10    Own Embeddings  0.615385
3              50          20          10  GloVe Embeddings  0.666667
4              50          30          10    Own Embeddings  0.782609
5              50          30          10  GloVe Embeddings  0.785714
6             100          10          10    Own Embeddings  0.500000
7             100          10          10  GloVe Embeddings  1.000000
8             100          20          10    Own Embeddings  0.666667
9             100          20          10  GloVe Embeddings  0.666667
10            100          30          10    Own Embeddings  0.700000
11            100          30          10  GloVe Embeddings  0.695652
12            200          10          10    Own Embeddings  0.666667
13            200          10          10  GloVe Embeddings  1.000000
14            200          20          10    Own Embeddings  0.615385
15            200          20          10  GloVe Embeddings  0.615385
16            200          30          10    Own Embeddings  0.727273
17            200          30          10  GloVe Embeddings  0.818182
```

For some hyperparameters, the F1 score of models trained by us was higher and for most the GloVe one was higher. The trends in Validation and Training loss show that it was overfitted to some extent (observed that the validation loss was decreasing and then increasing) which may result in the values of F1.

- Try with and without the pre-trained GloVe embeddings. Across your experiments, did the pre-trained embeddings work better, or was it better to train your own embedding layer for this task?

```
d = 200
h = 50
batch_size = 30
num_epochs = 10
learning_rate = 0.001
```

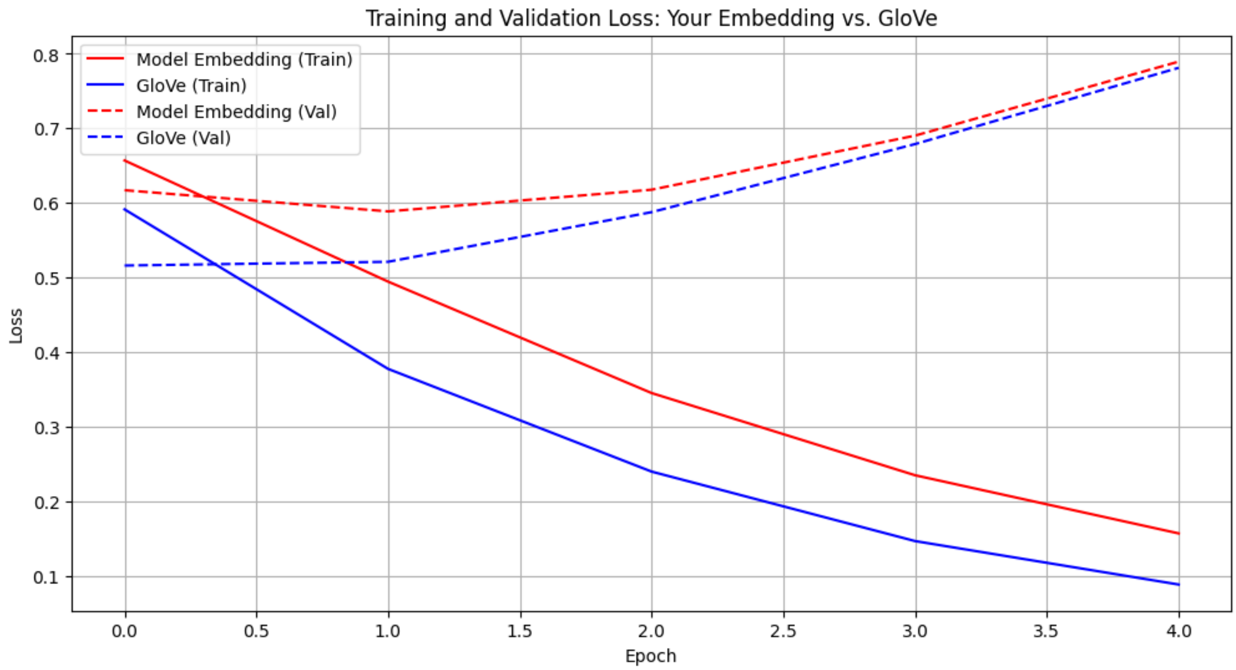
Trained the model for above parameters with and without GloVe embeddings:

```
Evaluation metrics on test data for without Glove embedding:
```

Accuracy: 0.7692
Precision: 0.7652
Recall: 0.7563
F1 Score: 0.7607

Evaluation metrics on test data for with Glove embedding:
Accuracy: 0.7795
Precision: 0.7592
Recall: 0.7988
F1 Score: 0.7785
0.7785108388312912

Comparison graph for losses:



- Did you notice any trends with certain parameters?
    - The GloVe embedding was working good for large values of embedding dimensions (d)
    - I had initially entered number of epochs as a parameter but then due to early stop it was not getting any further than 10, so removed that to decrease the for-loop time.
    - As the batch size was increased, the model was working better and better

- What happens if you freeze the GloVe embeddings during training time?

Tried the same parameters with freeze glove embeddings:
Evaluation metrics:
```
Accuracy: 0.7692
Precision: 0.7450
Recall: 0.7969
F1 Score: 0.7701
```

The F1 score for freeze=True was a bit less than the one with freeze=False

2. Compare your own embeddings, trained on the Rotten Tomatoes task, against the more general-purpose GloVe embeddings. Choose some words you think might be interesting: for example, since this is a dataset of movie reviews, the words “good” and “bad” might be interesting, since what is considered “good” in the movie domain may differ significantly from the word “good” more generally. Find the nearest neighbors for the words for both your own embeddings, and the GloVe embeddings.

Output:

```
Nearest neighbors for 'good' in own embeddings:
['window', 'greed', 'couture', 'veering', 'trilogy', 'jean', 'expands',
'preferably', 'kinnear', 'ghandi']
Nearest neighbors for 'bad' in own embeddings:
['dreamscape', 'goodness', 'execrable', 'symmetry', 'grocery', 'amish',
'booking', 'accent', 'freaking', 'alabama']
Nearest neighbors for 'good' in GloVe embeddings:
['always', 'you', 'better', 'way', 'excellent', 'well', 'really', 'get',
'we', 'going']
Nearest neighbors for 'bad' in GloVe embeddings:
['really', 'too', 'nothing', 'good', 'so', 'kind', 'going', 'wrong',
'think', 'awful']
```

What do you notice about the differences in the vector space between your own embeddings and the pre-trained embeddings?

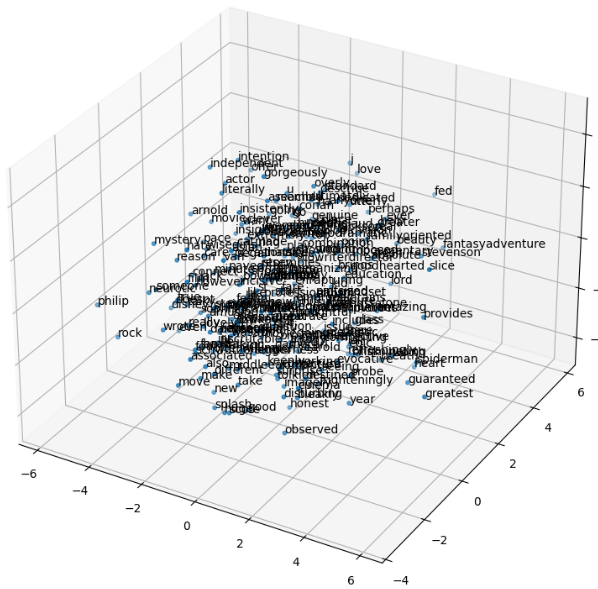
- For neighbors of word ‘good’ in our own trained model, there were words like couture, trilogy, veering which may have a context specific to movie reviews. Other words seem to be reference to characters or actions played in movies.
  On the GloVe side, the words are more sentimentally close to the word ‘good’. It shows the common words that are likely to appear with the word ‘good’

- For the word ‘bad’, words like symmetry, execrable, accent appear which may point towards the general way of giving movie reviews.
  GloVe embeddings on the other hand, gives us generic words like wrong, awful and also some antonyms like good, kind.

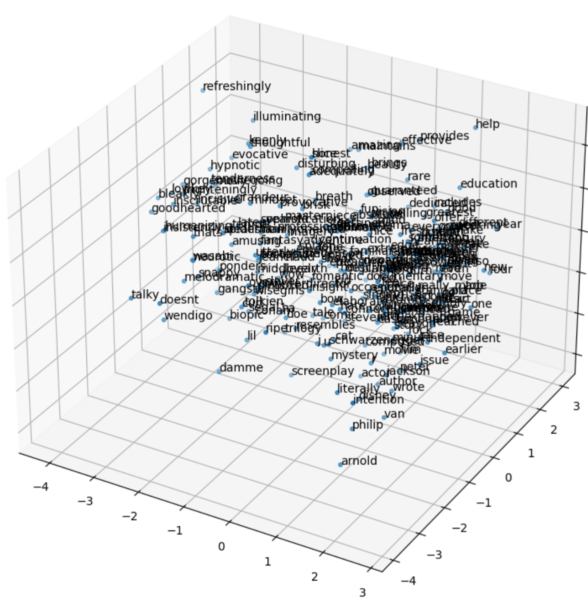
- Our models embedding captured more nuanced and context specific associations related to movie review. On the other hand, GloVe embeddings provide a more generalized sense of the given word.

3. See if you can make any interesting or illuminating visualizations of your own embeddings with the two plot_embeddings functions in the a3-explore-scaffolding.py file. If none of the visualizations make any sense to you, explain why you’re surprised by what you see, and what you would’ve expected instead. Speculate whether it’s because of the idiosyncrasy of the data, or if it’s because the embeddings haven’t been trained to optimally represent language, or something else entirely. Then, compare and contrast with visualiza- tions of the GloVe embeddings.

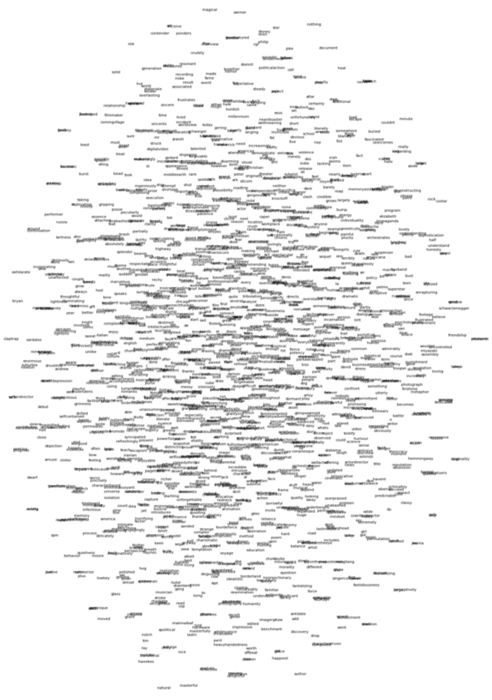
Without GloVe:



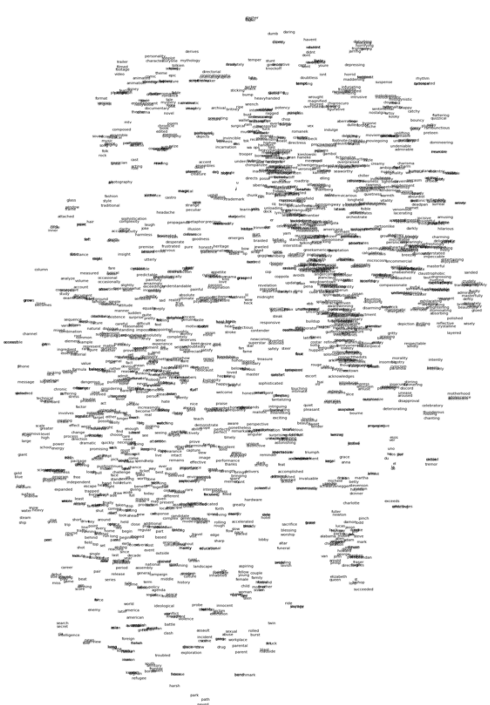
With GloVe:



Without Glove (tsne):



With Glove(tsne):



- Our models own embeddings show more domain-specific clusters, reflecting the language patterns observed in movie reviews.
- Also, its more scattered which may mean that our model was not trained that brilliantly. It might not have captured the more profound relationships between words. This could be due to limitations in the training data or its quality
- If a task requires more domain specific meanings to be captured, our model is better as compare to the Glove embeddings

- GloVe embeddings show more general semantic relationships, which can maybe be used for broader language tasks.
- GloVe embeddings show some clusters very close to each other, indicating that the word embeddings are more closely related in context.
- In a more generalized context of embeddings, using GloVe embeddings is more optimal.