NLP Assignment 2

Sharvita Paithankar

1.Describe the details for your best performing model: parameters such as number of epochs, batch size, and learning rate. Compare and contrast with other parameter settings you used—feel free to use graphs or tables to compare. Why do you think it performed better than the other models?

Change	Model	# of epochs	Batch size	Learning rate	Average Train loss	Average Test loss	F1 score (for test file)	Precision(for test file)	Recall (for test file)	Change observed
Base model	Figure 1	100	16	.1	0.4476227104663849	0.4211078087488104	0.862068965517 2413	0.7575757575757576	1.0	None since its the base model
Batch size	Figure 2	100	8	.1	0.3962582098810296	0.3846652865409851	0.877192982456 1403	0.78125	1.0	Batch size was reduced from 16 to 8 which decreased the train and test losses, This means that smaller batches helped the model generalize better. A smaller batch size helps for more gradient updates per epoch, which leads to better results than base model but also more noise.
# of epochs	Figure 3	50	16	.1	0.4901957541704178	0.4957176248232524	0.862068965517 2413	0.7575757575757576	1.0	The number of epochs was cut in half from 100 to 50. Both train and test loss increased, which means that the model did not train enough on the data and was underfitting. More

										training is needed for better results.
Learning rate	Figure 4	100	16	.5	0.3482585191726685	0.3476558476686477	0.892857142857 1428	0.806451612903	1.0	Learning rate was increased from 0.1 to 0.5. Both losses significantly decreased. But, a high learning rate is not good since it can cause overfitting, though in this case, it seems to have improved performance.
Learning rate	Figure 5	100	16	.01	0.6420522093772888	0.6460180282592773	0.8928571428571428	0.806451612903	1.0	Learning rate decreased from 0.1 to 0.01. Both losses increased, which means the model learned too slowly. This means that a too-small learning rate can prevent effective weight updates, and lead to bad results.
batch size	Figure 6	100	32	.1	0.5059190928936005	0.4547072499990463	0.8333333333333333	0.714285714285	1.0	Batch size increased from 16 to 32. Train loss increased, while test loss slightly decreased. This means that the larger batch size may have led to more stable updates but less adaptability to the test set. Large batch sizes help the model learn faster, but it might not explore enough of the data to find the best solution, leading to worse performance

										on new data. (possible overfitting)
# of Epochs	Figure 7	200	16	.1	0.3927773594856262	0.4346421758333842	0.892857142857 1428	0.8064516129032258	1.0	The number of epochs was doubled from 100 to 200. Train loss decreased, but test loss increased, meaning that there is probably overfitting. The model learned well on training data but started memorizing instead of generalizing, which led to a bad result on unseen data.

Best Performance:

The model in Figure 4 (Learning rate = 0.5) has the best performance, with the lowest train loss (0.3483) and test loss (0.3477). Increasing the learning rate to 0.5 helped the model to update its weights faster, which led to quicker learning. This also helped it get a better local minimum with the 100 epochs. Compared to Figure 5 with a learning rate of 0.01, which learned too slowly and resulted in higher losses, Figure 4 has a better balance between learning speed and gives a good result. . Figure 3 with 50 epochs, underfit due to fewer epochs, Figure 4 had 100 epochs, which allowed it to train enough but not too excessively, preventing overfitting (as seen in Figure 7 (200 epochs)).Overall, the model performed the best because the earning rate tuning was just perfect, not too high (causing divergence) or too low (learning too slowly). Also 100 epochs were enough for results without overfitting. A batch size of 16 also helped to avoid the downside of very small or very large batch sizes.

Here are the screenshots of each parameter:

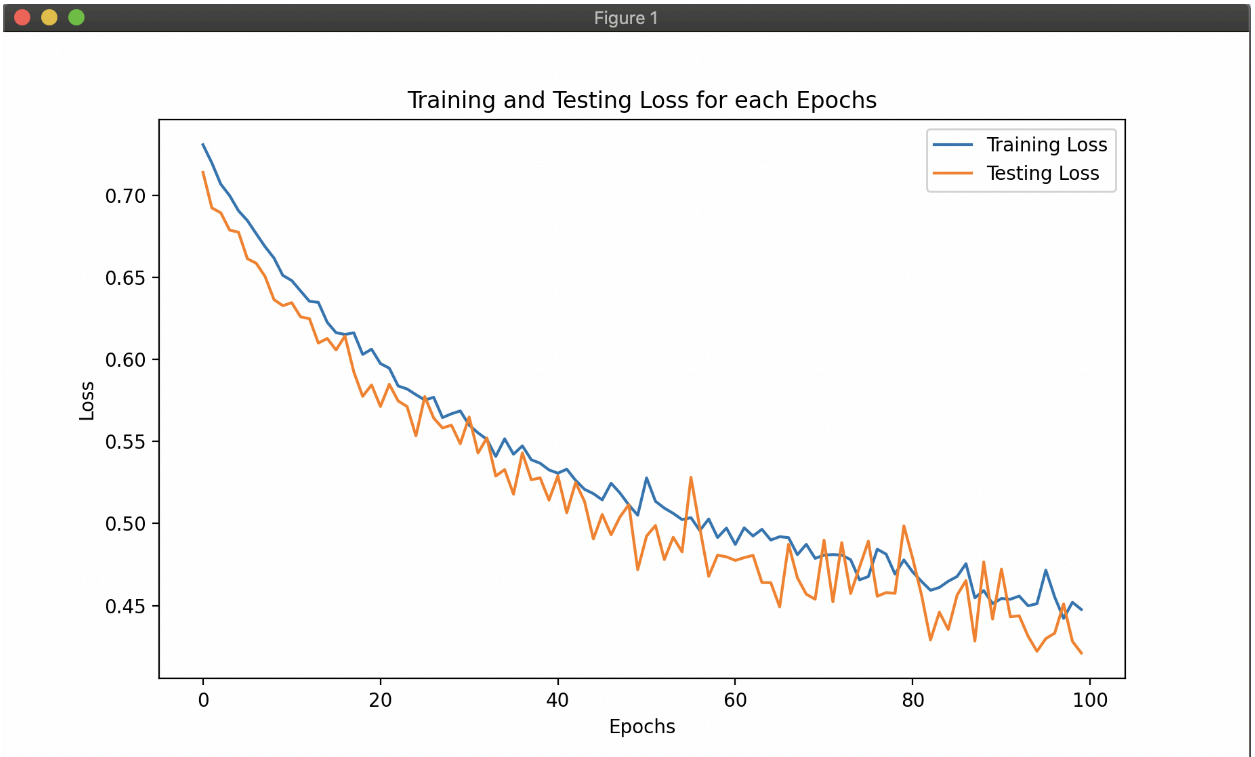
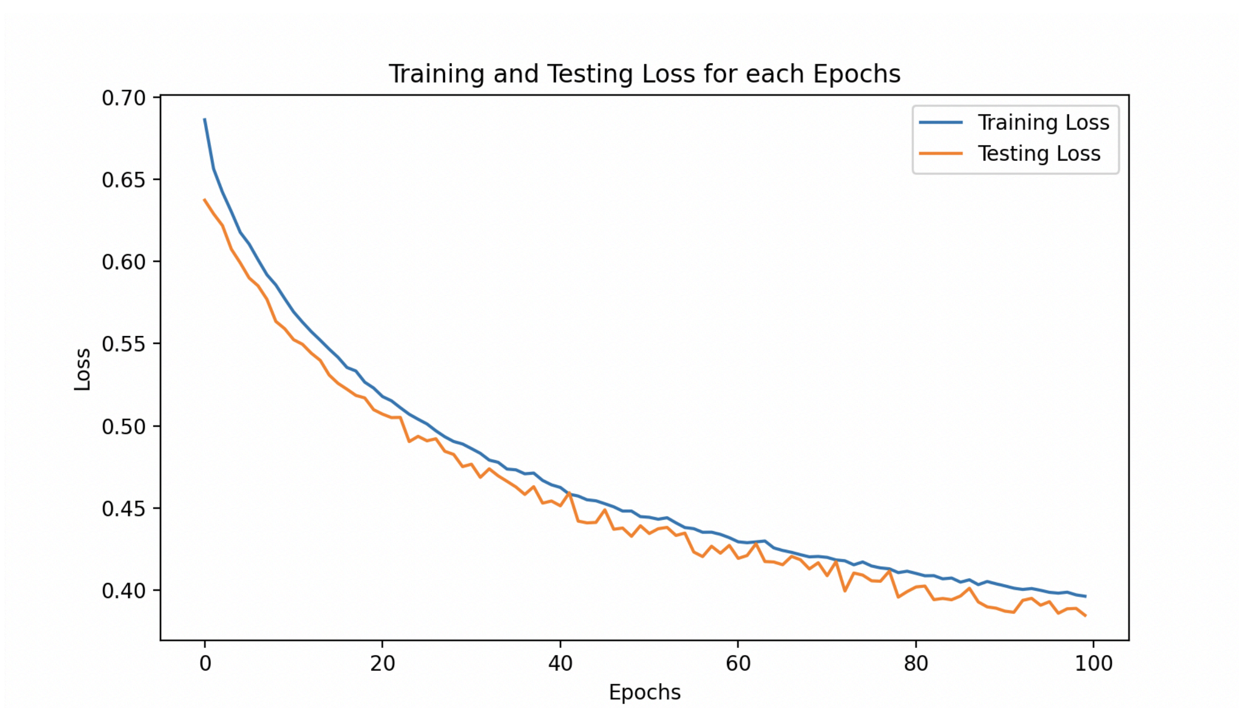


Figure 2



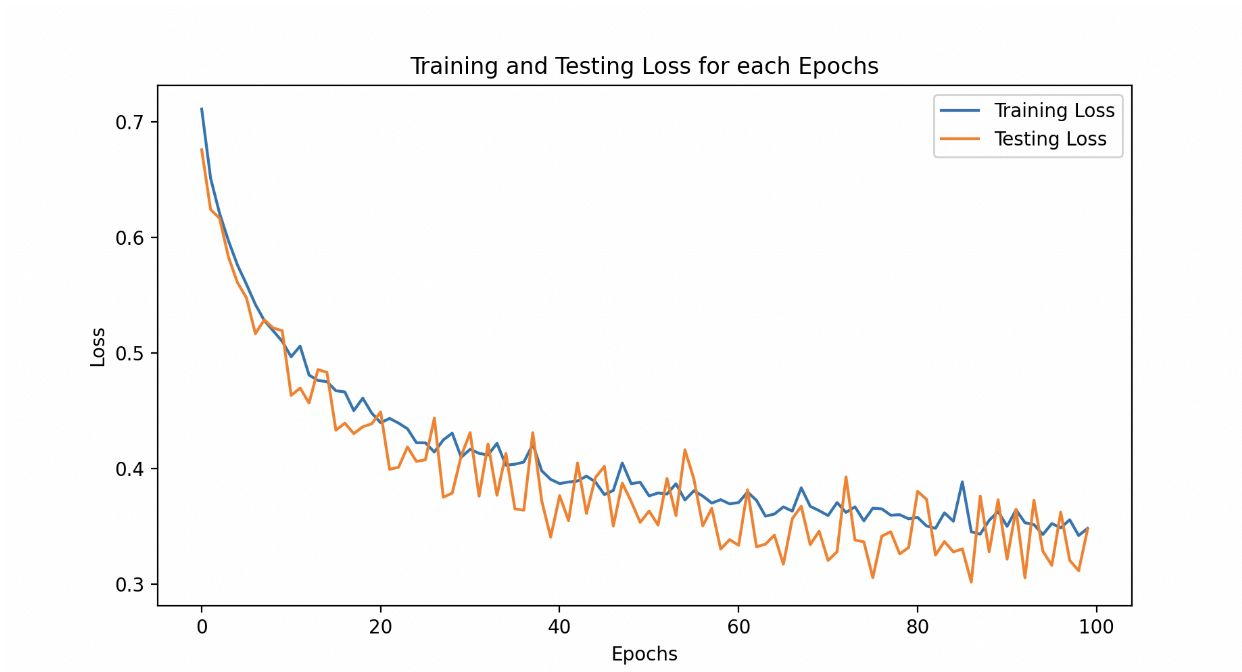
Epoch: 100
Avg train loss: 0.3962582098810296
Test Loss: 0.3846652865409851

Batch size: 8
Learning Rate: .1

Figure 3

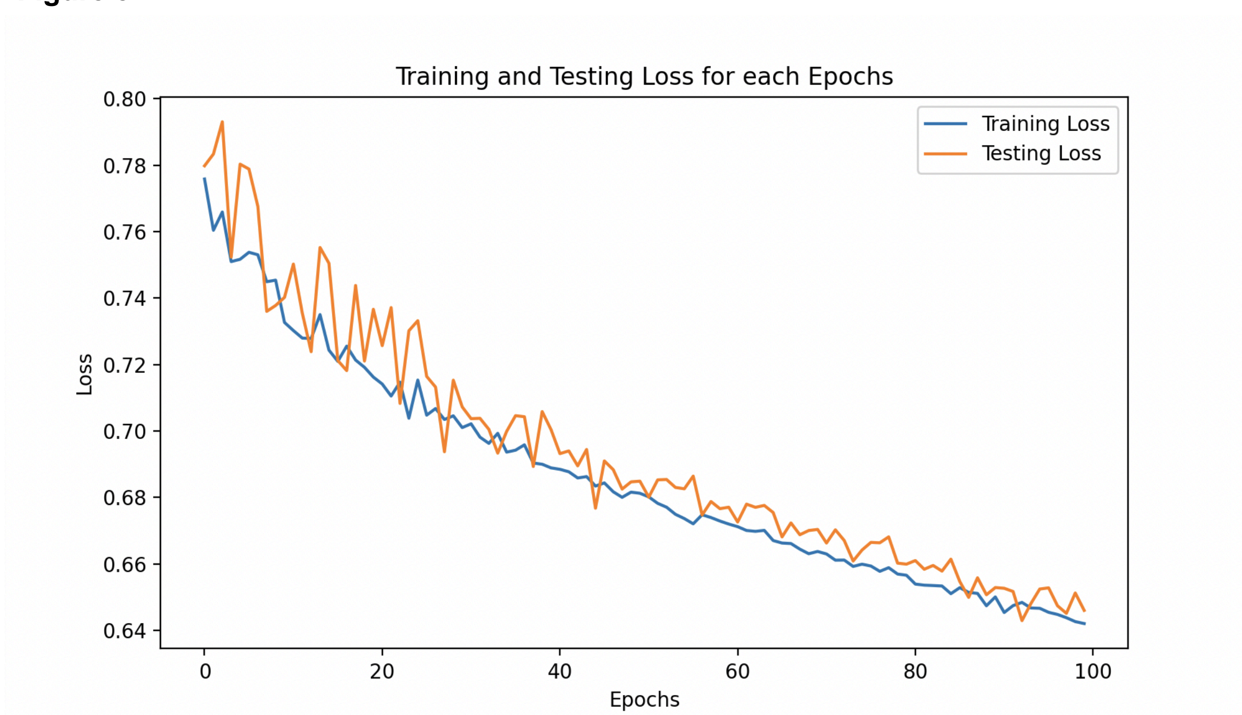
Epoch: 50
Avg train loss: 0.4901957541704178
Test Loss: 0.4957176248232524
Batch size: 16
Learning Rate: .1

Figure 4



Epoch: 100
Avg train loss: 0.3482585191726685
Test Loss: 0.34765584766864777
Batch size: 16
Learning Rate: .5

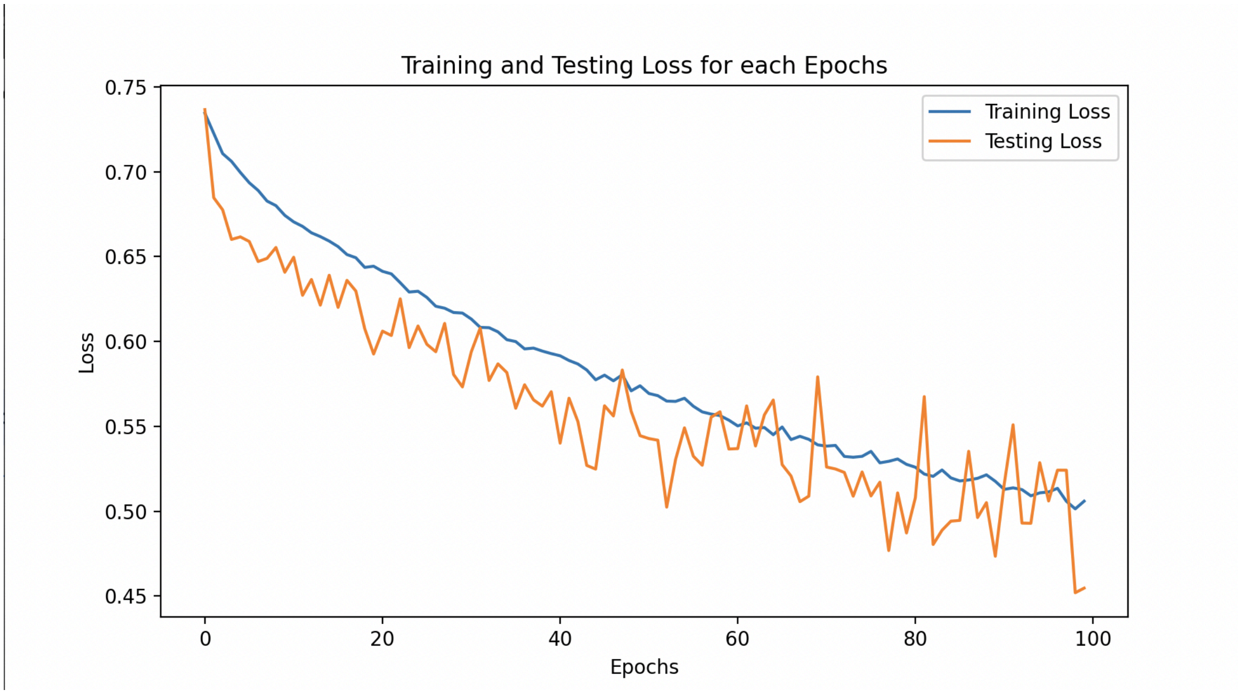
Figure 5



Epoch: 100

Avg train loss: 0.6420522093772888
Test Loss: 0.6460180282592773
Batch size: 16
Learning Rate: .01

Figure 6



Epoch: 100
Avg train loss: 0.5059190928936005
Test Loss: 0.4547072499990463
Batch size: 32
Learning Rate: .1

Figure 7

Epoch: 200
Avg train loss: 0.3927773594856262
Test Loss: 0.4346421758333842
Batch size: 16
Learning Rate: .1

2. What do you think would happen if you didn't normalize the feature vectors? Write down a guess for what you think would happen, and then run an experiment to test your intuitions and report back what you learned.

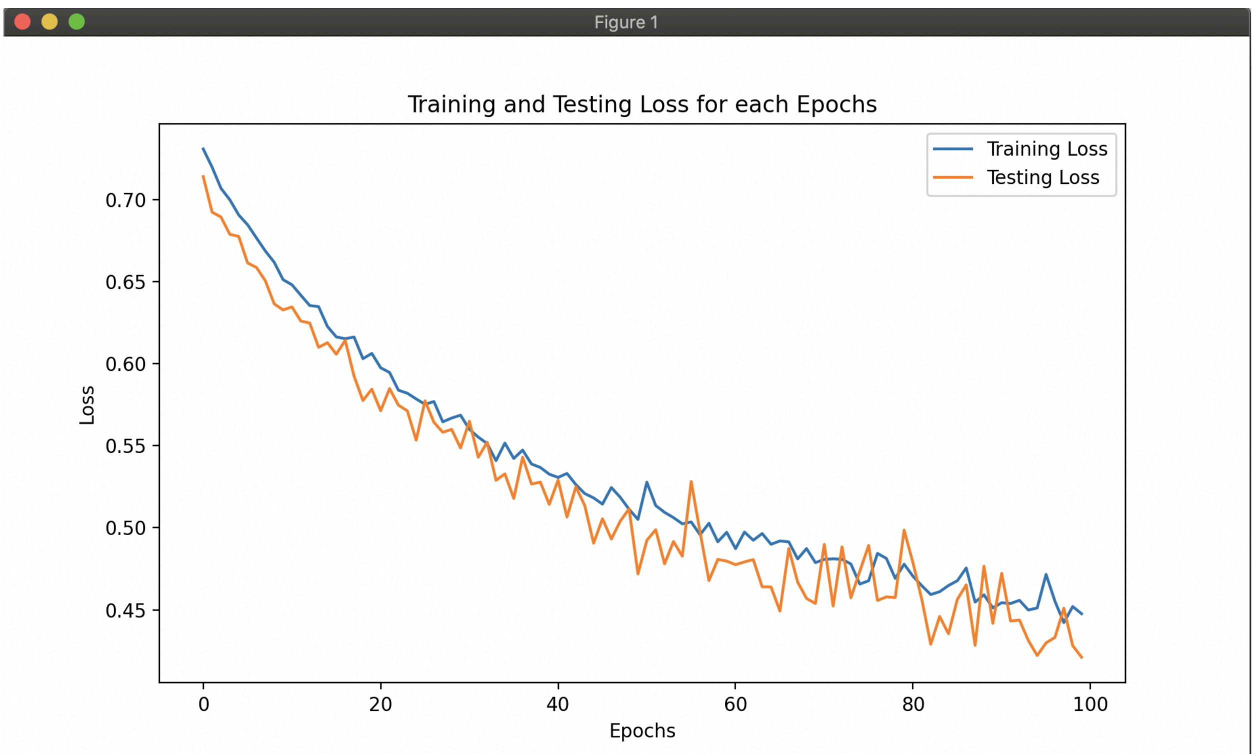
Expectations:
The model might not learn properly if we don't normalize the feature vectors. Some features might have much bigger numbers than others, making the model focus too much on the bigger numbers and ignore the smaller ones. This can cause the learning process to be unstable and make it slower or jumpy. The model might have trouble getting the best results which can then led to errors and poor performance on new data. When we remove the normalization, the model might overfit to certain features and fail to generalize well.

Tests:

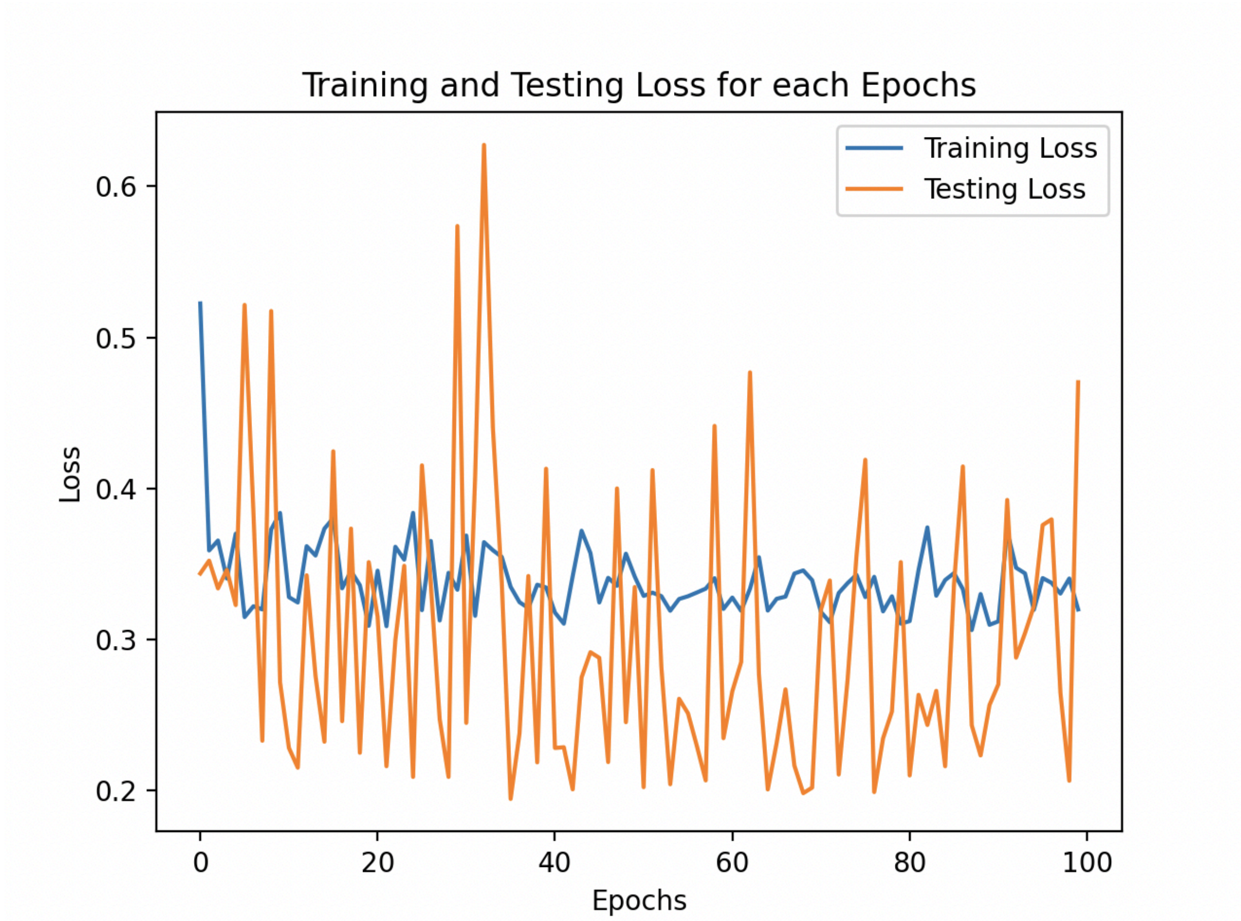
Change	# of epochs	Batch size	Learning rate	Average Train loss	Average Test loss	Change observed

Base model	100	16	.1	0.4476227104663849	0.42110780874888104	None since its the base model
Remove normalization	100	16	.1	0.3198295719921589	0.47025148073832196	When normalization was removed, the train loss decreased from 0.4476 to 0.3145, and the test loss decreased from 0.4211 to 0.2584.

Base model:



After removing normalization:



Outcomes:

The train and test loss both decreased when we removed normalization, which seems like an improvement at first but it's not because this might mean the model is overfitting and relying too much on features with larger values but ignoring smaller ones. Some features have much bigger numbers than others, the model may learn patterns too quickly but struggle to generalize to new data. Without normalization, the learning process can also become not stable. Some features might end up dominating which affects the gradient updates. Normalization helps keep all features balanced, making the model more reliable.

The graphs also help get a better understanding of how the model is learning and as you can see, before normalization (in figure 1), the model is learning at a good rate. The second graph shows that the learning curve is not good and all over the place. The testing loss fluctuated and had large spikes throughout and the model might be overfitting certain features. The spikes in the graph for test loss means that the model is not making smooth gradient updates and its happening because the model is not scaled properly. This is why normalization is important.

3. What would happen if you removed one of the features entirely and used a 5-dimensional feature vector? Choose one feature and remove it from your vector. Then, run another experiment and see what happens. Does the test F1 go up or down? Does the model converge slower, or faster? Report which feature you removed and what you Learned.

Expectations:

If we remove one feature and use a 5-dimensional feature vector instead of 6, F1 score might drop because of loss of useful information meaning the model will not have good predictions or if it was noisy, F1 might improve. With fewer features, the model might also train faster because less parameters need to be updated. Another thing we might notice is that the train/test loss might increase because an important feature was removed. Precision might remain the same if the model becomes more conservative in its positive predictions and recall might drop if the model loses classification power

---

Tests:

Change	**# of epochs**	**Batch size**	**Learning rate**	**Average Train loss**	**Average Test loss**	**F1 score**	**Recall score**	**Precision score**	**Change observed**
Base model	100	16	.1	0.4476227104663849	0.4211078087488 8104	0.9230769230769231	0.9473684210526315	0.9	None since its the base model
5-dimensional feature vector instead of 6	100	16	.1	0.5709549248218536	0.5793547431627909	0.6666666666666666	0.5263157894736842	0.9090909090909091	The decrease in recall while the precision stayed almost the same means the model became more conservative. The

									6-dimensional feature vector likely had important information for figuring out the positive cases The model was not able to figure out many positive cases and it can be seen through the recall but still has confidence in the cases it does identify

---

Outcomes

The change in F1, recall and precision means that the removed feature was important for the model performance, especially for identifying positive cases. The model became more conservative in its predictions and the decrease in F1 and recall scores means that it was a key feature for sentimental analysis.

4. Review Section 4.10, p. 18 of the textbook and then consider the resources we used for this task: for instance, the training data and the positive and negative lexicons. Did you notice any biases present in these resources? Can you think of any harms or unintended consequences (harmful or not) that this classifier could cause? There is no correct answer; just write a couple of sentences reflecting on this prompt.

Yes there are biases present in this in the resources. One of the biases on the resource used for this task is the fact that we are relying on a specific dataset which is the hotel reviews and the positive and negative lexicons are pre-defined. This could potentially cause an issue because let's say a word like "quirky" can be used in a positive and a negative way. A hotel review could have the word "quirky" in a positive way while another review could have the word in a negative way. This bias can led to misclassification and cannot be a reliable sentiment classification. It could either make or break a business and hence, could be “harmful”.