INLP Assignment 3 Report

Name - Pranav Gupta

Roll No. - 2021101095

As a general thumb of rule for the entire assignment, Sequence Length to train the LSTM for Downstream Classification Task was taken to be the length which was greater than 95 percent of the samples in the train set. This value came out to be 59. Learning Rate of 1e-3 and HIdden Dimension size = 256 were taken to train the LSTM. To train the LSTM, train set was divided in batches of 100. LSTM was trained for 10 epochs for both SVD and Skip-Gram Models. Also, for Hyperparameter Tuning for both the Models (SVD and Skip-Gram), Embedding Size was taken to b 50 instead of the usual 300 across the assignment. Also, as a general rule, for Skip-Gram, Embeddings were trained for 5 epochs.

5. Analysis:

SVD:

Hyperparameters used to train the Model-

1. Window Size = 2 was used to consider the positive samples and target word pairs and then was used to train the model.
2. Embedding Size = 300 was used to represent each word.
3. All the parameters of LSTM were taken as specified above.

Evaluation Metrics obtained -

1. Train Set:

   Training Set Accuracy = 82.25166666666667

   Recall = 82.25166666666667

Precision = 83.21663927915915

F1-Score = 82.23682537651327

Average Training Loss = 0.004908875096589327

2. Test Set:

   Testing Set Accuracy = 72.75

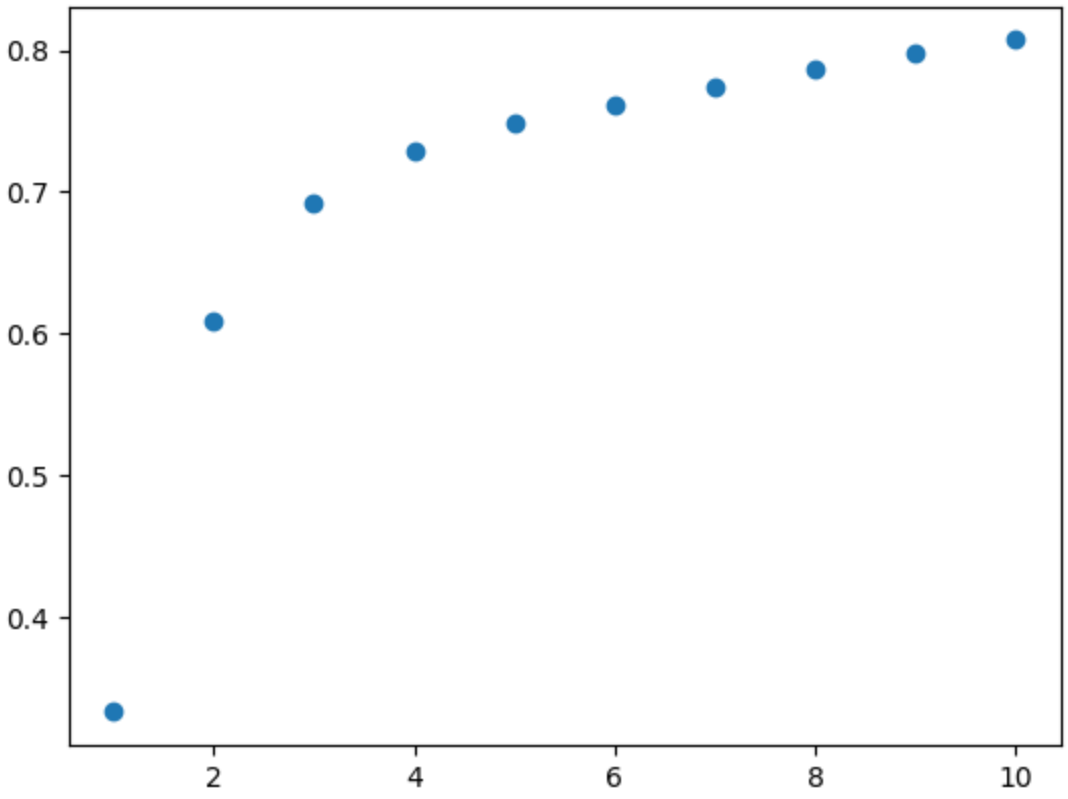
   Recall = 72.75

   Precision = 73.1093351800554

   F1-Score = 72.88439959010665

   Testing Loss = 0.00010358908912166953

   Confusion Matrices obtained for both Training as well as Test Datasets are present in the Jupyter Notebook attached.



Training Set Accuracy Vs Number of Epochs in Downstream Classification Task

Skip-Gram Model:

Hyperparameters used to train the Model-

1. Window Size = 2 was used to consider the positive samples and target word pairs and then was used to train the model.
2. Embedding Size = 300 was used to represent each word.
3. All the parameters of LSTM were taken as specified above.

Evaluation Metrics obtained -

1. Train Set:

   Training Set Accuracy = 97.9225

   Recall = 97.92250000000001

   Precision = 98.06252809409584

   F1-Score = 97.91947912282164

   Average Training Loss = 0.0006389037007465959

2. Test Set:

   Testing Set Accuracy = 88.5

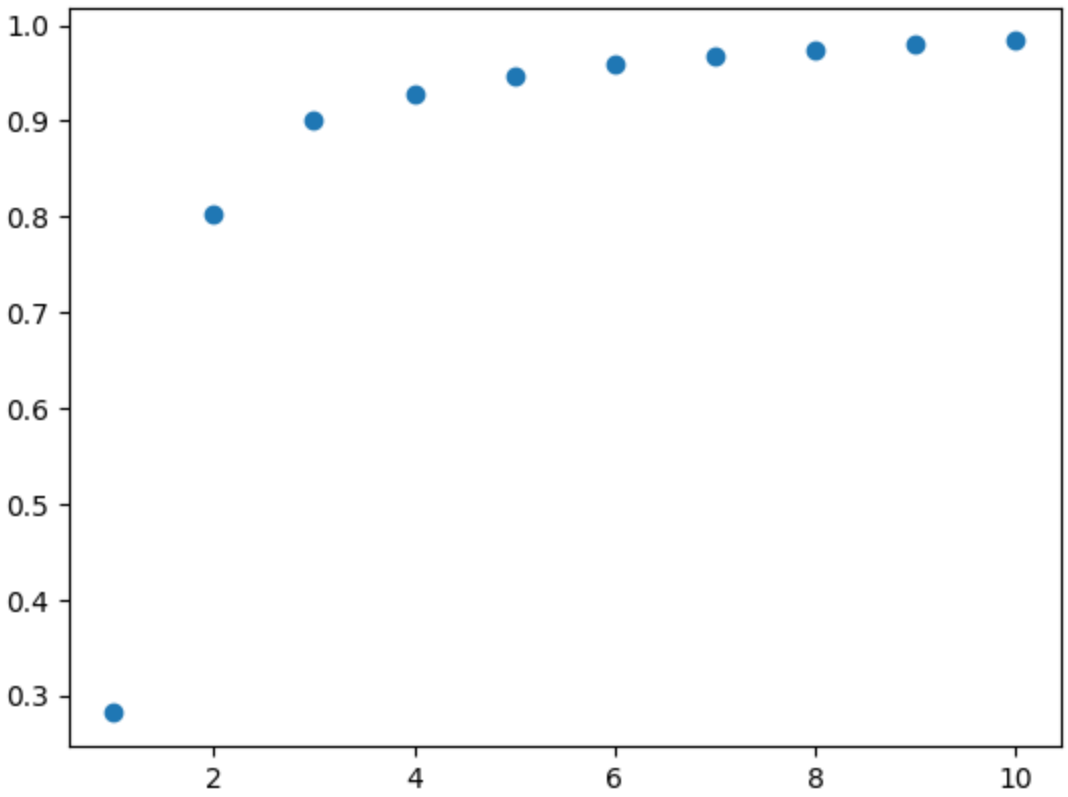
   Recall = 88.5

   Precision = 88.68131578947368

   F1-Score = 88.52638169111162

   Testing Loss = 6.855813262518495e-05

   Confusion Matrices obtained for both Training as well as Test Datasets are present in the Jupyter Notebook attached.



Training Set Accuracy Vs Number of Epochs in Downstream Classification Task

Results -

We observe that in general Word2vec (Skip-Gram Model with Negative Sampling) performs better than SVD in all the metrics mentioned above as specified in the Assignment PDF. It is because of the following reasons:

1. Skip-Gram is trained using LSTM, which learns to represent words based on their contexts in the corpus. In contrast, SVD treats words as independent entities and may not capture the subtle semantic relationships between them as effectively.
2. Skip-Gram considers the context in which words appear in the training data, capturing information about word co-occurrences and relationships within a window of context words. On the other hand, SVD treats each word in isolation and may not capture contextual information as effectively.
3. Skip-Gram learns distributed representations for words based on their contexts, allowing it to capture meaningful representations even for words with sparse occurrences in the training data. SVD, on the other hand, may

struggle to generate accurate embeddings for rare words due to the sparsity of their occurrences in the matrix being decomposed.

Shortcomings:

SVD has a number of drawbacks while being good at capturing latent semantic information and lowering dimensionality. To begin with, its word embeddings can be memory and computationally intensive, especially for big vocabularies. Furthermore, because SVD considers words individually and ignores contextual information, its embeddings may be less sophisticated and less able to capture semantic linkages depending on usage context. Furthermore, because unusual words appear infrequently in the training set, SVD may have trouble accurately capturing them.

Skip-Gram is one of the Algorithms in Word2Vec Model. Word2Vec is particularly good at extracting semantic relationships and contextual information. Furthermore, Word2Vec models are domain-specific, which means that without further fine-tuning on domain-specific data, they might not function as well in specialised fields. Large volumes of text data are also needed for Word2Vec model training, which presents difficulties for applications in specialised fields or languages with limited resources. The selection of context window size during training can have a substantial impact on the quality of Word2Vec embeddings.

Hyperparameter Tuning:

Hyperparameters Settings used to train the Model-

Window Size = 1, 2, 3 were chosen to perform the experiment to understand the importance of context window size for Classification. These Window Sizes were chosen to reduce the time taken to train the Model for Training of Embeddings while on the other hand, also understanding the impact of understanding the impact of context windows. As more and more samples are taken into consideration in the context windows, model is able to generalise better which samples perform better as positive and negative samples which on the other hand allows the Model to capture the dependencies of the Words involved in the Text.

Since, the number of words in context window in the 3 cases specified above are 3, 5 and 7 respectively, we can still notice the trends of learning the embeddings effectively and make suitable comparisons because the Training Dataset is huge and more or less the embeddings of all the samples will be adjusted accordingly in the dataset. Also, embedding dimension of 50 was taken to draw comparisons. All parameters of LSTM were taken as specified above.

SVD:

Evaluation Metrics:

Context Window Size = 1

1. Train Set:

   Training Set Accuracy = 66.07333333333333

   Recall = 66.07333333333333

   Precision = 67.95405105338129

   F1-Score = 65.98017809919337

   Average Training Loss = 0.008467479608952999

2. Test Set:

   Testing Set Accuracy = 58.53947368421053

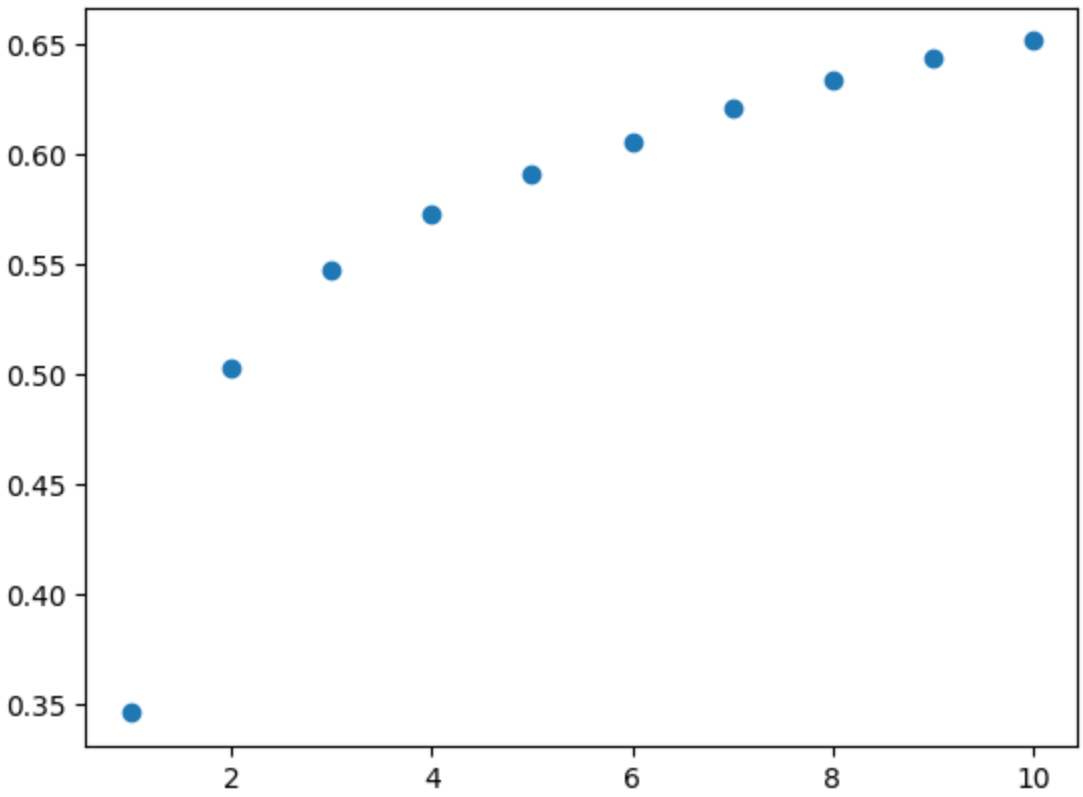
   Recall = 58.53947368421053

   Precision = 59.09096952908587

   F1-Score = 58.66057942402126

   Testing Loss = 0.0001357786386506632

Confusion Matrices obtained for both Training as well as Test Datasets are present in the Jupyter Notebook attached.



Training Set Accuracy Vs Number of Epochs in Downstream Classification Task

Context Window Size = 2

1. Train Set:

   Training Set Accuracy = 65.93083333333334

   Recall = 65.93083333333333

   Precision = 67.65449766940893

   F1-Score = 65.7648546112857

   Average Training Loss = 0.008505964651703835

2. Test Set:

   Testing Set Accuracy = 57.27631578947369

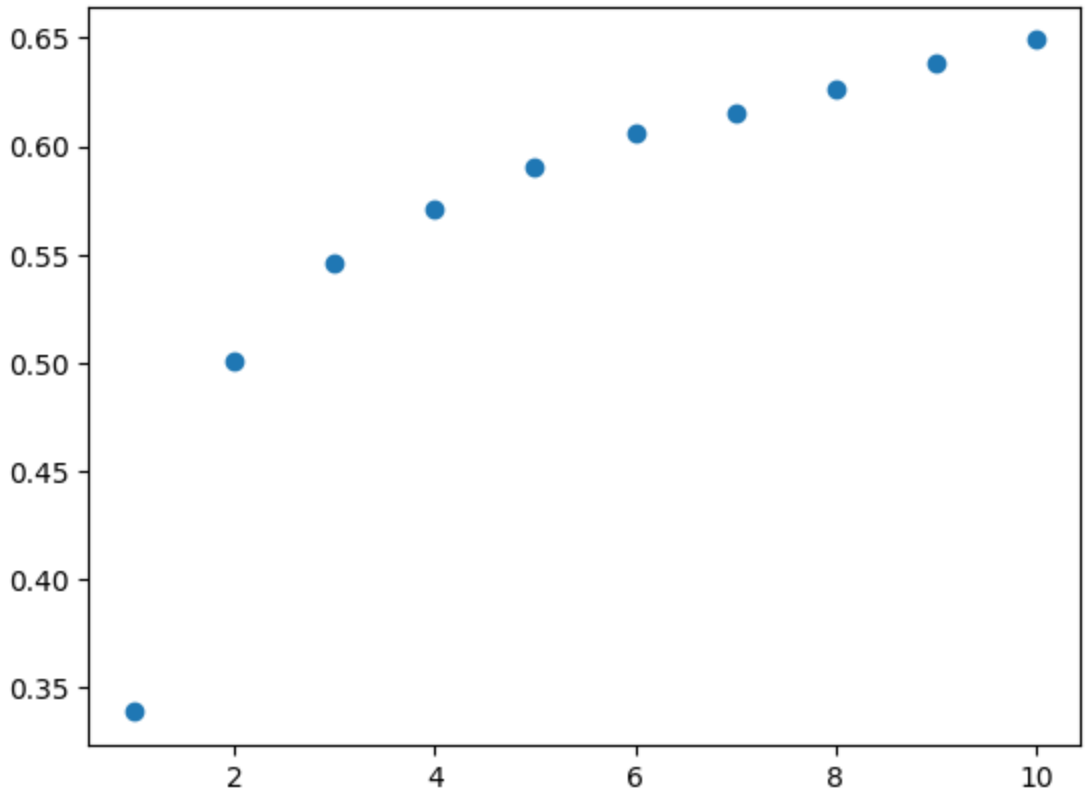
   Recall = 57.27631578947369

   Precision = 58.05228531855956

F1-Score = 57.44343303162504

Testing Loss = 0.00013605097774416208

Confusion Matrices obtained for both Training as well as Test Datasets are present in the Jupyter Notebook attached.



Training Set Accuracy Vs Number of Epochs in Downstream Classification Task

Context Window Size = 3

1. Train Set:

   Training Set Accuracy = 64.29833333333334

   Recall = 64.29833333333332

   Precision = 65.64437601825045

   F1-Score = 64.03924984721228

Average Training Loss = 0.008844300173223019

2. Test Set:

   Testing Set Accuracy = 56.21052631578948

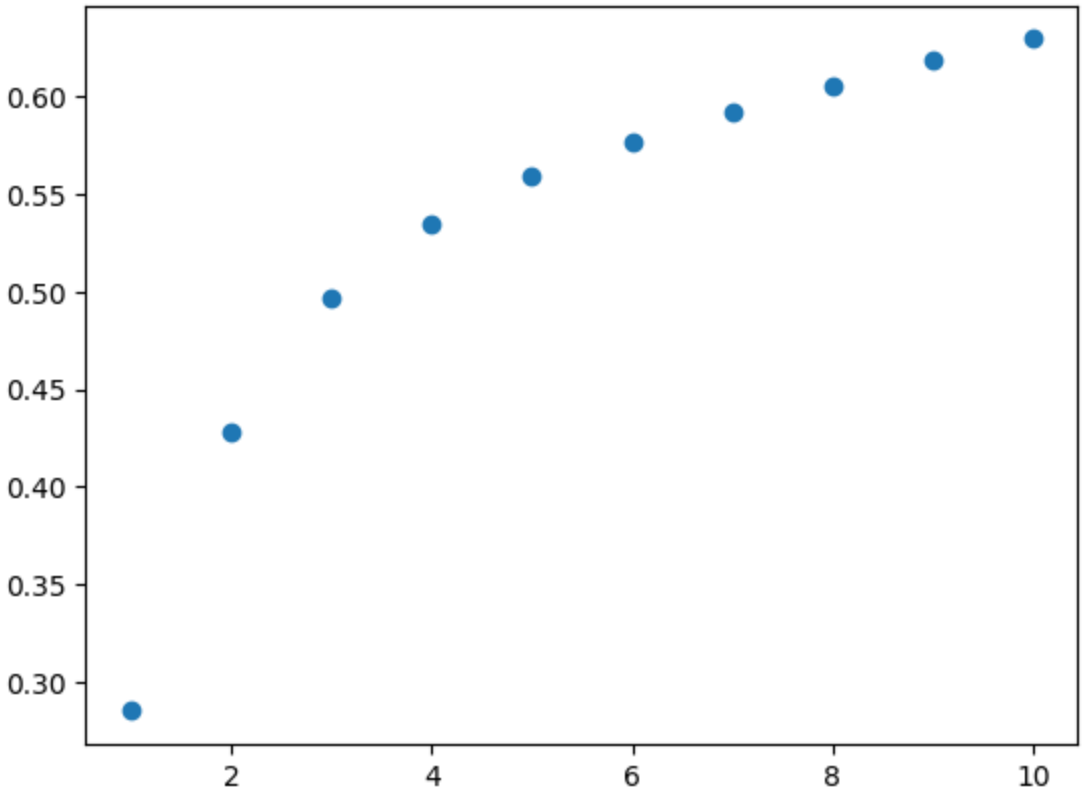
   Recall = 56.21052631578948

   Precision = 56.68663434903047

   F1-Score = 56.32476957910101

   Testing Loss = 0.00014017483044881374

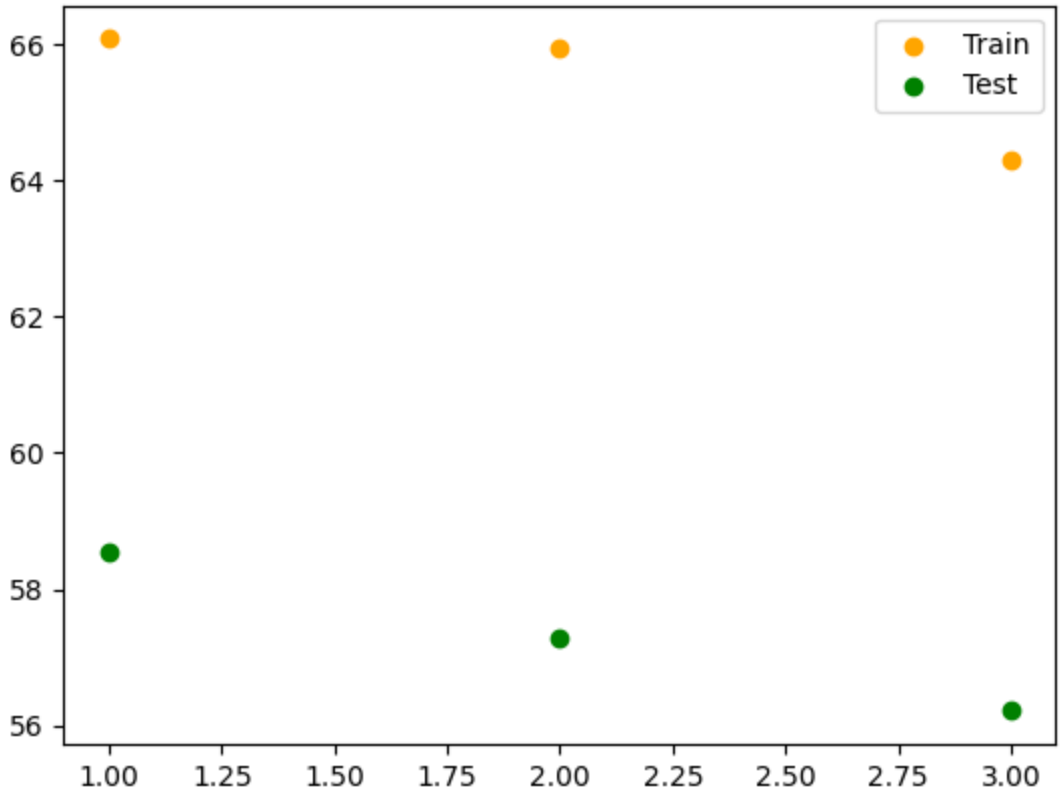
Confusion Matrices obtained for both Training as well as Test Datasets are present in the Jupyter Notebook attached.



Training Set Accuracy Vs Number of Epochs in Downstream Classification Task

We observe that all the 3 configurations perform almost similar with minor fluctuations. It is because the Training Dataset is so huge that the Model,

regardless of whether the context size is big or small is able to learn the context of a particular word. On account of this, our LSTM is able to capture the dependencies exactly, leading to almost similar results and performance metrics.



Comparison of train set and test set accuracies by varying size of context windows.

Skip-Gram:

Evaluation Metrics:

Context Window Size = 1

1. Train Set:

   Training Set Accuracy = 92.91833333333334

Recall = 92.91833333333334

Precision = 93.43379087297012

F1-Score = 92.91042257423574

Average Training Loss = 0.0021735006012022495

2. Test Set:

   Testing Set Accuracy = 85.98684210526316

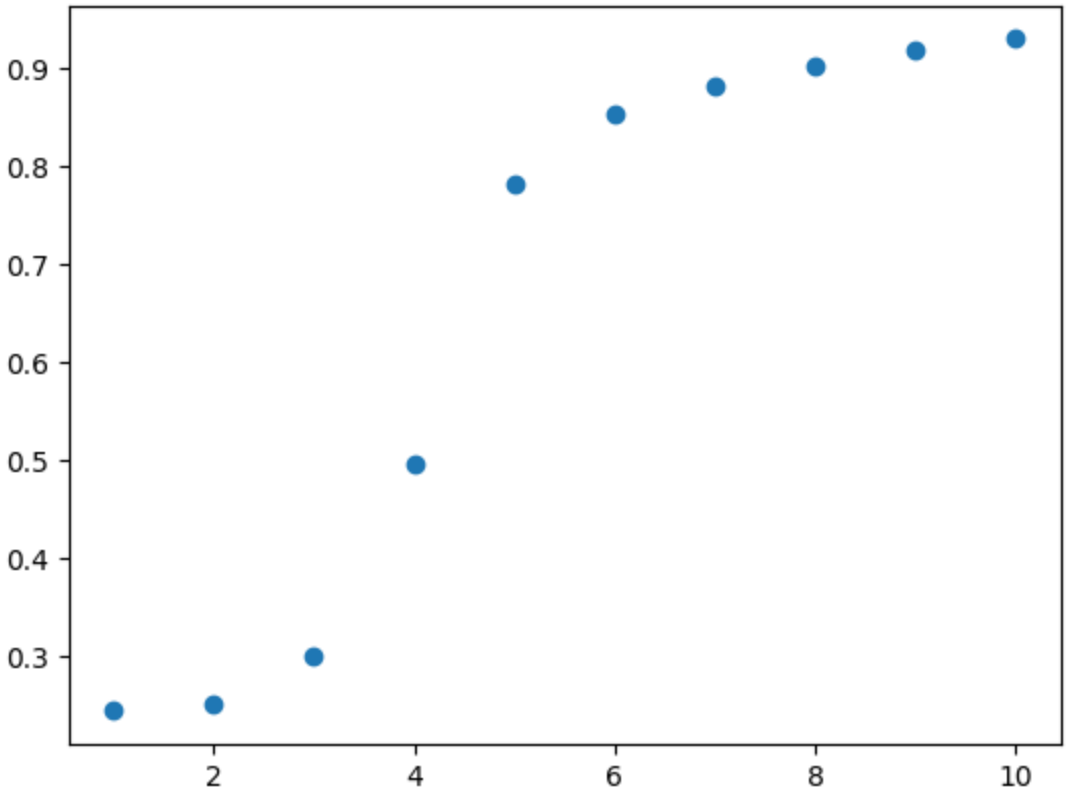
   Recall = 85.98684210526316

   Precision = 86.47009695290859

   F1-Score = 86.08927130970534

   Testing Loss = 5.9401521866675466e-05

Confusion Matrices obtained for both Training as well as Test Datasets are present in the Jupyter Notebook attached.



Training Set Accuracy Vs Number of Epochs in Downstream Classification Task

Context Window Size = 2

1. Train Set:

   Training Set Accuracy = 93.00333333333334

   Recall = 93.00333333333333

   Precision = 93.6421206606781

   F1-Score = 93.01234429207267

   Average Training Loss = 0.0021732973400503397

2. Test Set:

   Testing Set Accuracy = 85.68421052631578

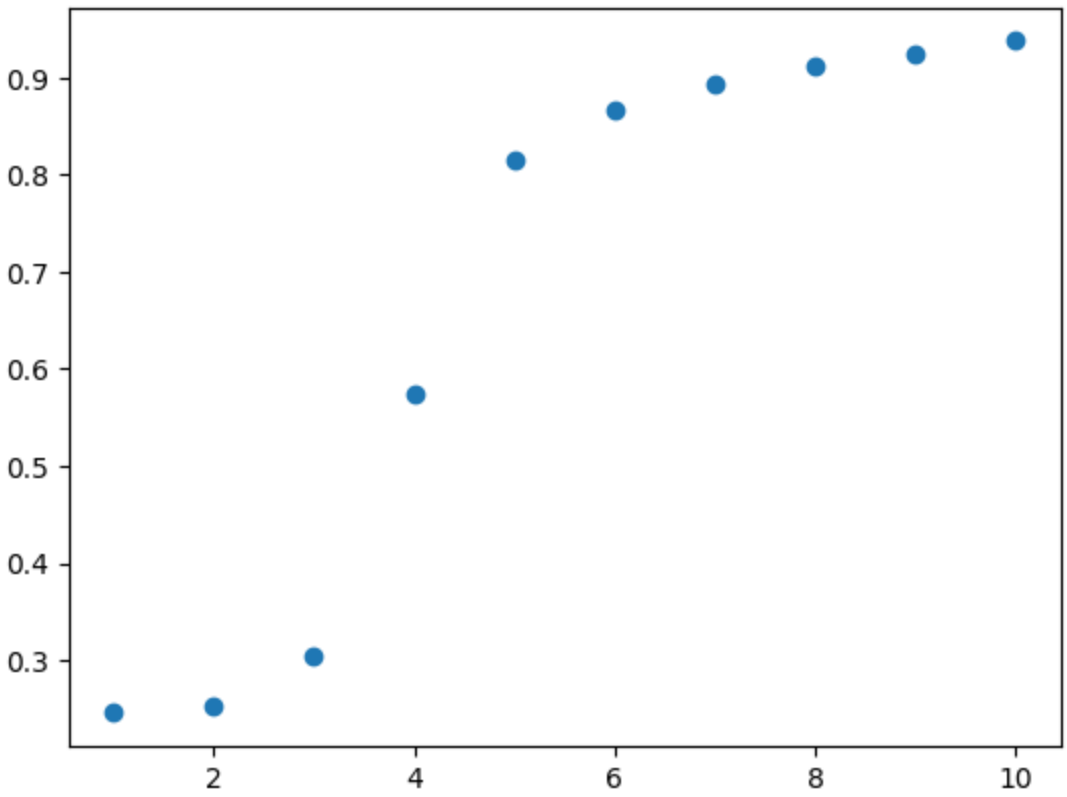
   Recall = 85.68421052631578

   Precision = 86.38512465373962

   F1-Score = 85.81816337286763

   Testing Loss = 6.968576781218871e-05

Confusion Matrices obtained for both Training as well as Test Datasets are present in the Jupyter Notebook attached.



Training Set Accuracy Vs Number of Epochs in Downstream Classification Task

Context Window Size = 3

1. Train Set:

   Training Set Accuracy = 90.70583333333333

   Recall = 90.70583333333334

   Precision = 91.36482083274072

   F1-Score = 90.67620177278076

   Average Training Loss = 0.002820103894919157

2. Test Set:

   Testing Set Accuracy = 85.81578947368421

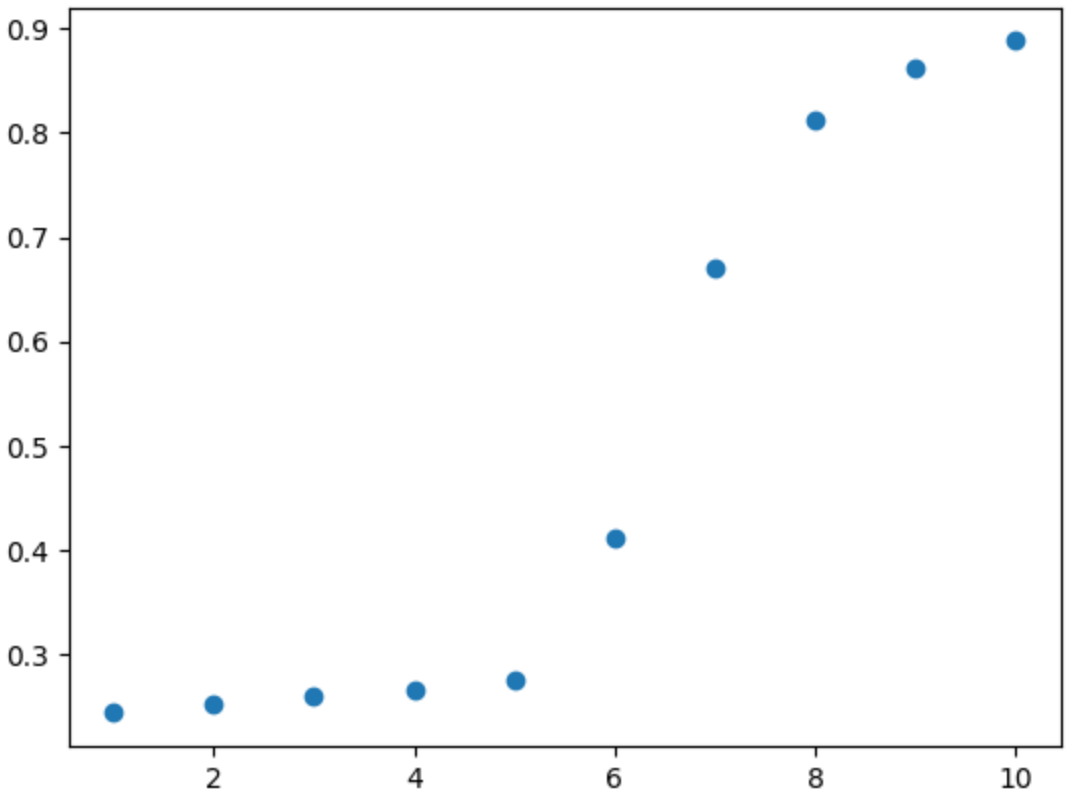
   Recall = 85.81578947368421

Precision = 86.1314404432133

F1-Score = 85.8353144400639

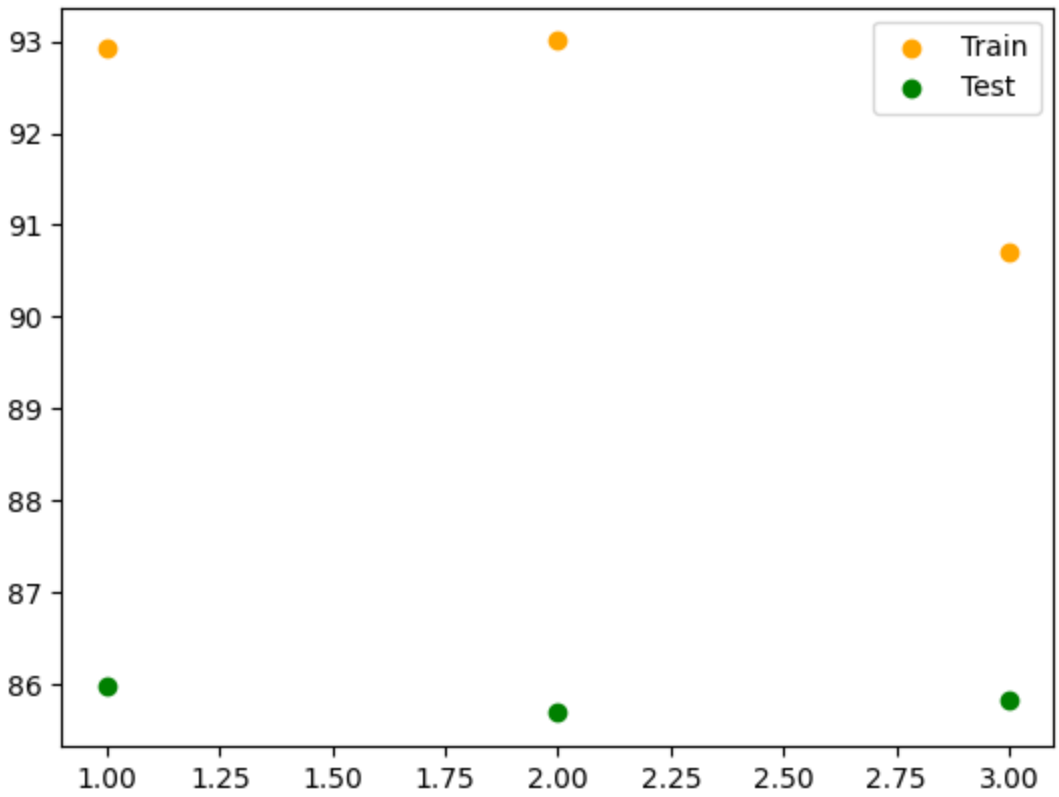
Testing Loss = 5.600050644716248e-05

Confusion Matrices obtained for both Training as well as Test Datasets are present in the Jupyter Notebook attached.



Training Set Accuracy Vs Number of Epochs in Downstream Classification Task

We observe that all the 3 configurations perform almost similar with minor fluctuations. Same reasoning goes here as well as was given in case of SVD. The Training Dataset is so huge that the Model, regardless of whether the context size is big or small is able to learn the context of a particular word. On account of this, our LSTM is able to capture the dependencies exactly, leading to almost similar results and performance metrics.



Comparison of train set and test set accuracies by varying size of context windows.

One more thing can be observed from above. We notice that accuracies are severely impacted as embedding size is reduced from 300 to 50 in case of SVD while in case of Skip-Gram, they only decrease slightly. This is a clear indication of the fact that SVD takes each of the word as a separate entity and then creates embeddings and completely ignores the neighbouring context while on the other hand, even when the embedding dimension is reduced for skip-gram, because of the consideration of the context, Skip-Gram model is able to learn the Word Vector representations very efficiently. This is one of the important reasons why skip-gram performs better than SVD in general.