Intro to Natural Language Processing

ELMO: Deep Contextualised Word Representations

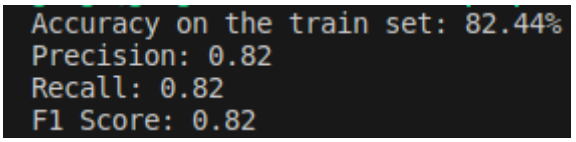
ASSIGNMENT 4

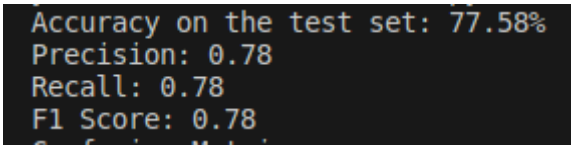
1. Analysis

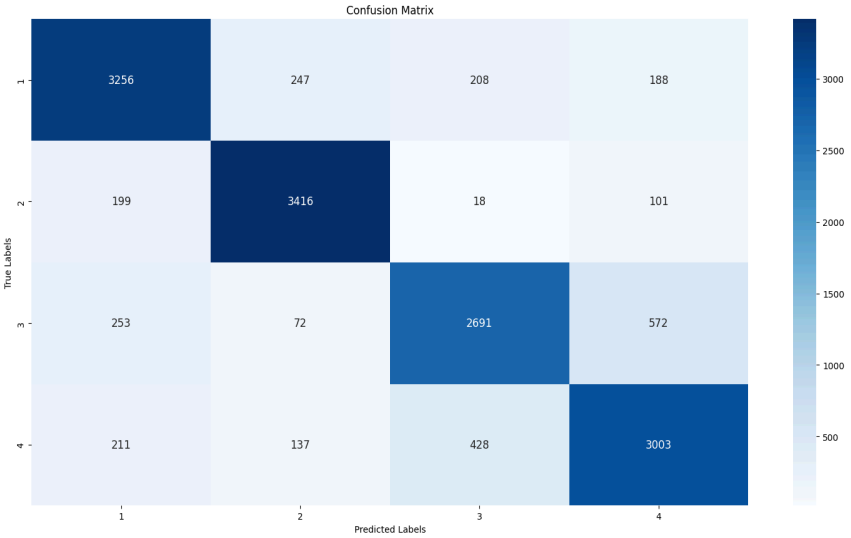
SVD is a dimensionality reduction technique used to generate word vectors by reducing the dimensionality of the co-occurrence matrix. Whereas, skip-gram helps to obtain a meaningful, low-dimensional representation of a word which is an aggregation of all possible contexts that the word has appeared in. However, both of these techniques fail to represent the word in different ways when it takes different meaning according to the context.
ELMO generates word embeddings based on the entire sentence in which the word occurs and thus is able to capture contextual information to provide embeddings which vary according to the context of the word.

In the last assignment, we performed the downstream task of classification using the word embeddings generated from SVD and Skip-Gram with Negative Sampling. The results on the testing and training data for the task are as follows -

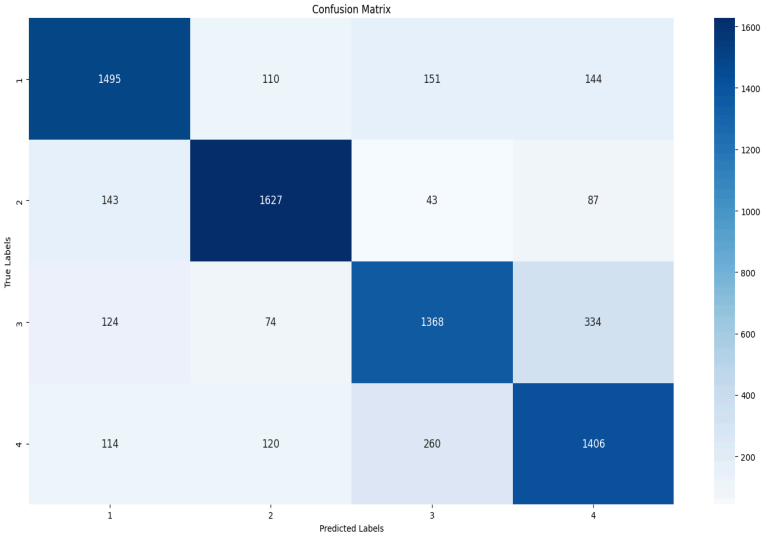
a. Singular Value Decomposition (SVD)







Confusion Matrix for SVD word vectors on training data

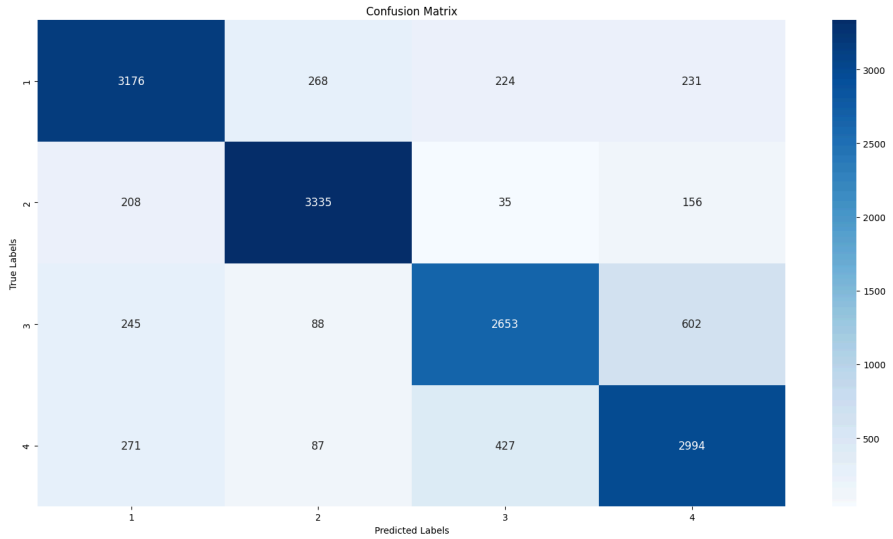


Confusion Matrix for SVD word vectors on testing data

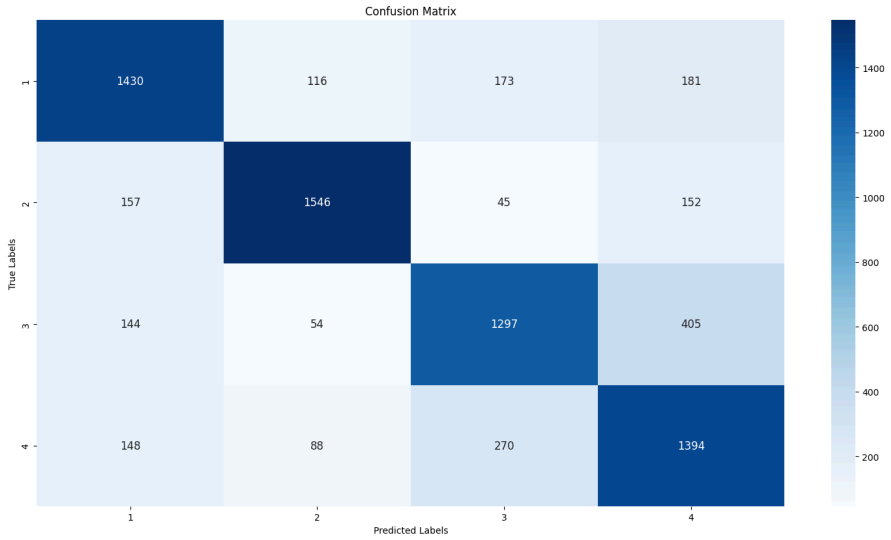
b. Skip Gram with Negative Sampling (word2vec)

```
Accuracy on the train set: 81.05%
Precision: 0.81
Recall: 0.81
F1 Score: 0.81
```

```
Accuracy on the test set: 74.57%
Precision: 0.75
Recall: 0.75
F1 Score: 0.75
```



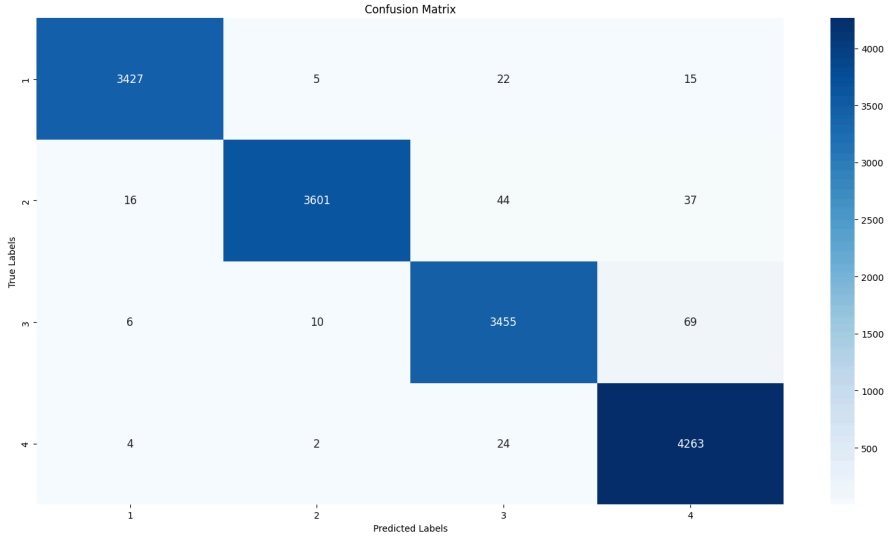
Confusion Matrix for Skip-Gram with Negative Sampling word vectors on training data



Confusion Matrix for Skip-Gram with Negative Sampling word vectors on testing data

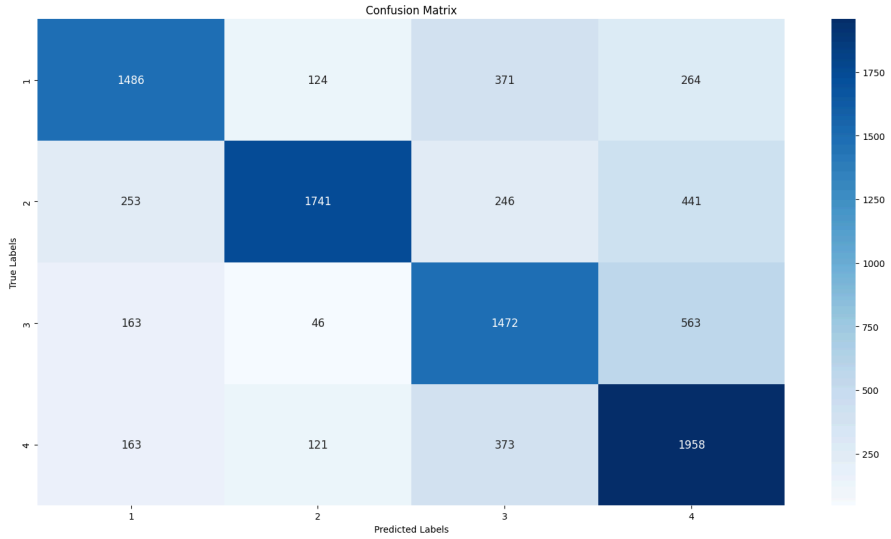
In this assignment, we built an ELMO architecture for generating word embeddings. We used pre-trained word embeddings from skip-gram. For the ELMO architecture we use a two-layered Bi-LSTM in this task.

The values of various evaluation metrics using the ELMO word embeddings on the downstream task are as follows -




```
Accuracy on the test set: 98.31%
Precision: 0.98
Recall: 0.98
F1 Score: 0.98
```

Confusion Matrix and Accuracy on Training Data




```
Accuracy on the test set: 68.26%
Precision: 0.69
Recall: 0.68
F1 Score: 0.68
```

Confusion Matrix and Accuracy on Testing Data

It can be observed that ELMO outperforms SVD and Skip-Gram with Negative Sampling in the downstream tasks in the training set due to its ability to capture contextual information. They allow for better representation of word meanings in different contexts.

However for the Testing data, it was observed that ELMO vectors show a slightly lower performance in comparison to SVD and Skip Grams, the reason could be - **Overfitting the training data** (the model which we trained might have overfitted the training data, leading to high performance on the training set and low performance on test set). Another reason could be **Pruning of Training Data** (while the testing data has more than 9.5K sentences the training data has been cut down and contains only 15K sentences which is not sufficient for the model to be trained for unseen data).

2. Hyperparameter Tuning

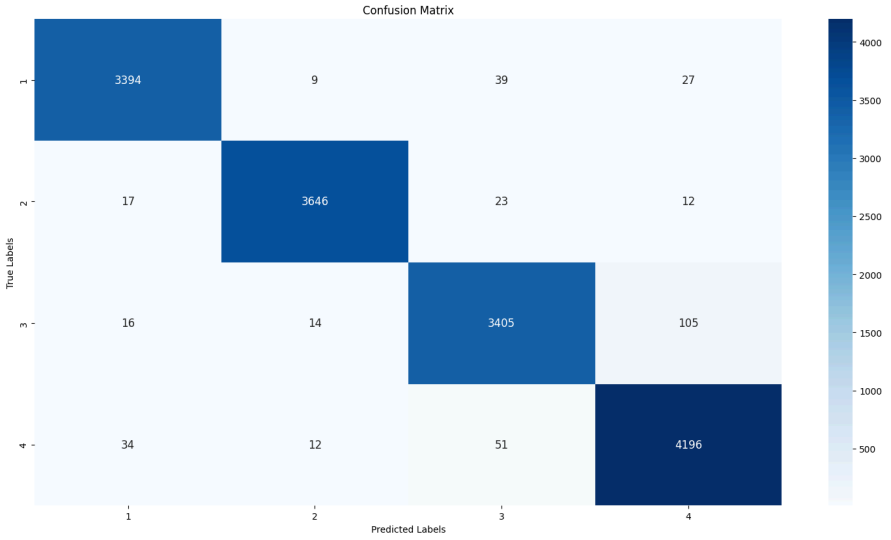
For the downstream classification task, following values of different hyperparameters were used -

- num_epochs = 50
- Learning Rate = 0.005
- Embedding Dimension = 128
- Hidden Dimension = 256
- Batch Size = 50

For combining word representations across the different layers of the biLSTM in the downstream task, we used λs. The following Hyperparameter settings were tried for -

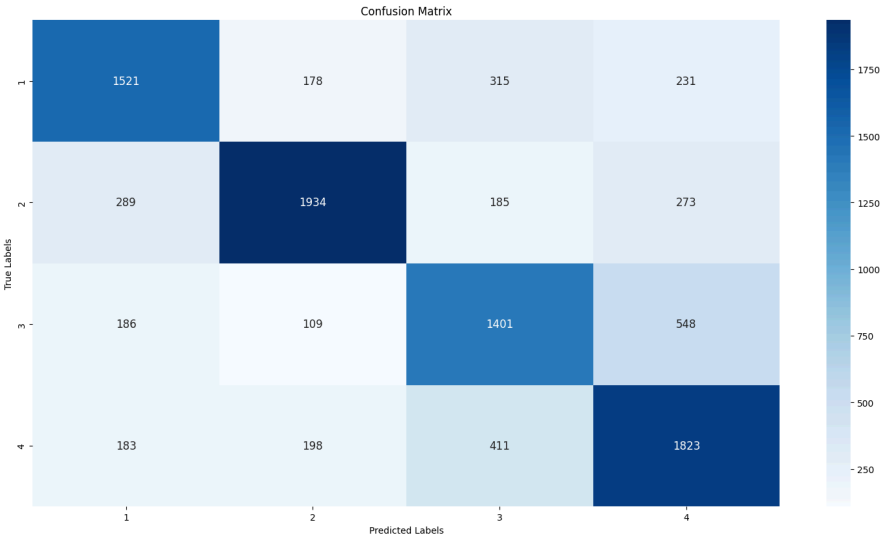
a. Trainable λs

In this setting we trained the model to find the best λs, and then used these for combining word representations across different layers.
The values of various evaluation metrics in this setting is as shown -




```
Accuracy on the test set: 97.61%
Precision: 0.98
Recall: 0.98
F1 Score: 0.98
```

Confusion Matrix and Accuracy on Training Data

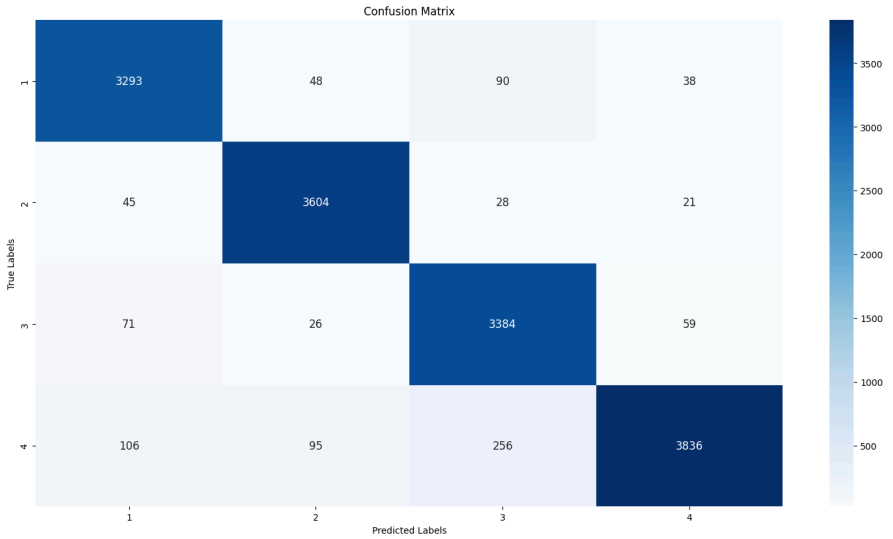


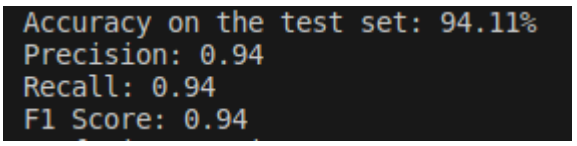

```
Accuracy on the test set: 68.03%
Precision: 0.70
Recall: 0.68
F1 Score: 0.68
```

Confusion Matrix and Accuracy on Testing Data

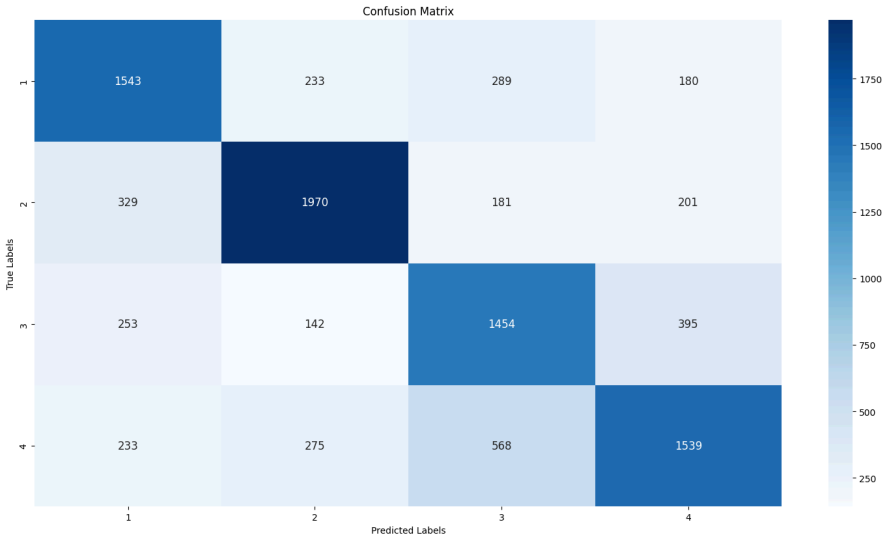
b. Frozen λs

In this setting we randomly initialised and then freezed the λs, (they were not changed during the training for the downstream classification task) The values of various evaluation metrics in this setting is as shown -





Confusion Matrix and Accuracy on Training Data

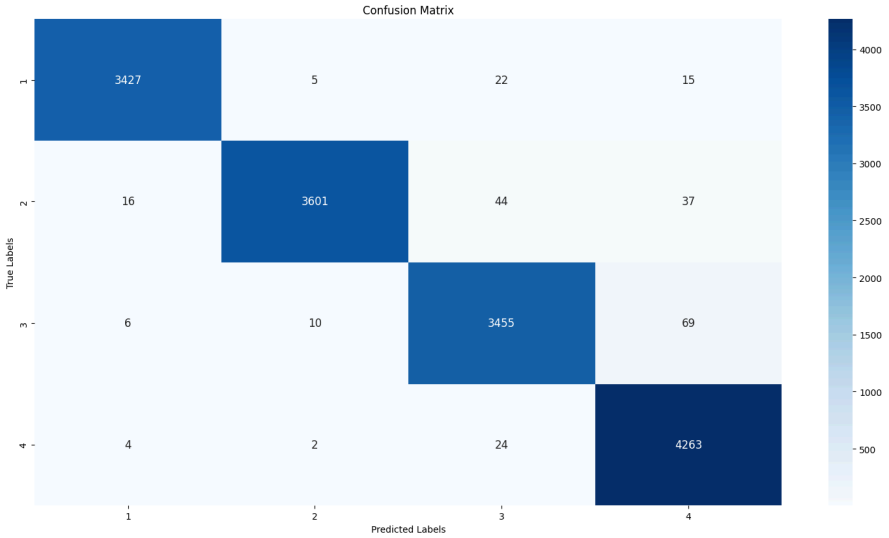


```
Accuracy on the test set: 66.49%
Precision: 0.67
Recall: 0.66
F1 Score: 0.67
```

Confusion Matrix and Accuracy on Testing Data

c. Learnable Function

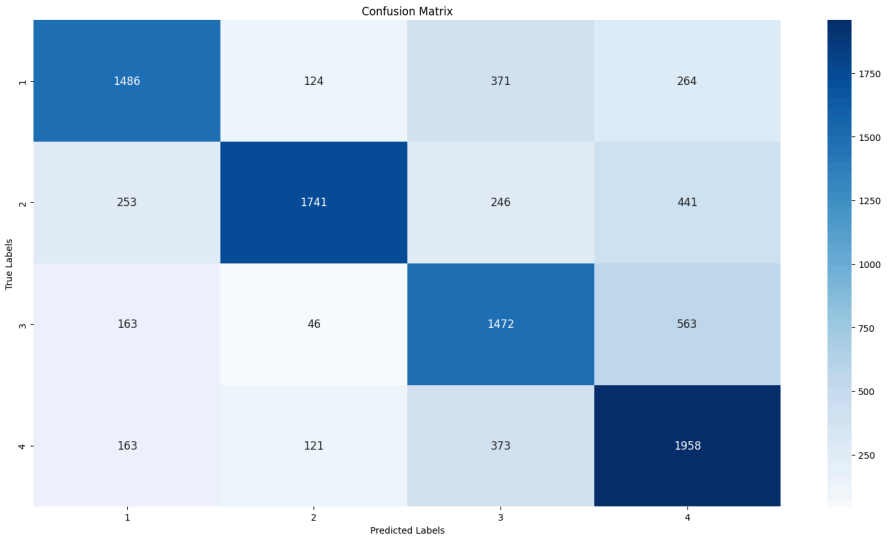
In this setting we learnt a function to combine the word representations across different layers. The values of various evaluation metrics in this setting is as shown -




```
Accuracy on the test set: 98.31%
Precision: 0.98
Recall: 0.98
F1 Score: 0.98
```

Confusion Matrix and Accuracy on Training Data

```
Accuracy on the test set: 68.26%
Precision: 0.69
Recall: 0.68
F1 Score: 0.68
```



Confusion Matrix and Accuracy on Testing Data

It can be observed that the learnable function proved to be the best setting for choosing the value of lambda's across different layers. The reasons could be -

- Frozen lambda values are initialised randomly (or manually) and they remain fixed throughout the training process, hence they lack task specific optimization.
- Trainable lambda's do provide some flexibility but they are still limited to linear combinations of word representation which may not be sufficient.
- The learnable functions on the other hand provides flexibility by learning a function to best capture relevant features from all the layers and it can be optimised according to the task performed.