MULTI-LAYER PERCEPTRON: 1-

layer MLP, 2-layer MLP, Random Forest and Logistic Regression.

TABLE OF CONTENTS

1. INTRODUCTION

The dataset given for this assignment consists of 50K Movie Reviews involving binary sentiment information (positive or negative) for each review. The final aim of this task was to compare at least three different models at predicting the sentiment of each review in the test dataset. The models that were implemented are: a 1-layer multi-layer perceptron (MLP) using pre-trained word embeddings, a 2-layer MLP using TF-IDF vectorizer, a Logistic Regression and a Random Forest classifier. The evaluation obtained in the four models was measured according to the accuracy of predicting the test data.

The first step was to clean the textual data, by removing stop-words, capitalization, HTML tags, and non-alphabetic characters, including punctuation. Moreover, WordNetLemmatizer was used to lemmatize verbs, adjectives and nouns, with the final aim to reduce the number of features. By applying all these cleaning steps to the data, the vocabulary dataset consisted of 153,603 words. The next step was to convert the categorical features (negative or positive sentiment) to labels (‘0’ or ‘1’) using LabelEncoder.

1.1 TWO-LAYER MLP

In the two-layer MLP neural network, the textual data was vectorised using TF-IDF (Term Frequency–Inverse Document Frequency). TF-IDF defines the importance of a term by taking into consideration the importance of that term in a single document, and scaling it by its importance across all documents. With the main purpose of reducing the number of vocabulary, min_df=10 was used. This hyperparameter is used to ignore terms that have a frequently strictly lower occurrence than the given threshold. In this way, the original number of words, which accounted for 153,603 was reduced to 5,283. Finally, this vectorizer was fit on the data by mapping the words onto the index values. It is also noteworthy to mention that because TF-IDF is based on the bag-of-words (BoW) model, it does not capture the semantics.

Because the IMDB.csv document did not contain the 25K training reviews separated from the 25K testing reviews, the data was randomly split. However, instead of using half of the data for testing, it was decided that only 20% (10,000 samples) will be used as test data, with the final aim to obtain better accuracy.

Subsequently, the following step was to apply TensorDataset to convert the arrays to tensors. Likewise, DataLoader was used to access the dataset into batches of 64. Dataloader is usually used to iterate through the data, manage batches and transform the data.

The activation functions implemented in this model, to generate non-linear neural networks, were: Sigmoid and Rectified Linear Units **(**ReLu), which are used for binary classification tasks (e.g., classifying positive and negative reviews). ReLU is *non- linear* and different from the Sigmoid function, it has the advantage of not having any backpropagation errors. Furthermore, for larger Neural Networks, the speed of building models based off on ReLU is very fast as opposed to using Sigmoids.

In the first layer, the input features (number of words) were 5,283, and the output dimension 100. Consequently, the input dimensions in the second layer were 100, and the output features 1 (‘0’ = negative, ‘1’ = positive).

To avoid overfitting, dropout regularization was applied. Dropout is a computationally inexpensive but powerful regularization method. In this regularization technique, the nodes are randomly selected and dropped to make a learning model more generalized so that it will perform better on newly arrived data. The dropout regularization was set to 0.5.

ADAM was adopted as the optimizer of this model to minimize the loss function by finding the optimal weights for each node. Its learning rate was set to 1e-3. On the other hand, the loss function of choice was binary cross-entropy (BCE), since the classification problem can be reduced to a binary choice (negative or positive). The concept emphasizes that if the loss is reduced to an acceptable level, the model indirectly learns the function that maps the inputs to the outputs.

After training the model for five epochs, the test accuracy accounted for 88,16% and the test loss was 0.295. See *Image 1.*

```
Epoch: 05 | Epoch Time: 6.63
Train Loss: 0.160 | Train Acc: 93.81%
Test Loss: 0.295 | Test. Acc: 88.16%
```

Image 1. Results obtained from the two-layer MLP model

1.2 ONE-LAYER MLP

In the one-layer MLP neural network, words were converted to word embeddings using Word2Vec, with the main purpose of maintaining the word context through meaningful numerical representations. Word2Vec is a neural network model that learns word associations. It is considered to be one of the biggest, most recent breakthroughs in the field of Natural Language Processing (NLP). Therefore, to convert words to word embeddings, Google news Word2Vec pre-trained model was used (1.5GB). This pre-trained model includes word vectors for a vocabulary of 3 million words and phrases that have been trained on roughly 100 billion words from a Google News dataset. The vector length is 300 features, which means that each word will be represented by 300 dimensions. To download word2vec-google-news-300, 'gensim.downloader' was used.

The first step was to tokenize the words from the textual reviews (vocabulary size = 153,603), and transform each text into a sequence of integers or indexes. Subsequently, the sequences were padded at the end (padding='post') to ensure that all of them had the same length. This process consists of adding '0s' at the end of each sequence until each sequence has the same length as the longest sequence (being the maximum length 1437).

The data was again randomly split, with the test data accounting for 20%. Similarly, the arrays were converted to tensors, following the same process as before, with the batch size being equal to 64. Likewise, the Sigmoid function, BCE and Adam optimizer with its learning rate set to 1e-3, were applied.

Regarding the dimensions in the one-layer MLP, the input dimensions were represented by the vocabulary size (153,603), meanwhile, the hidden dimensions were set to 512 and the output dimensions were equalled to 1 (1=positive, 0=negative). Moreover, as mentioned previously, the dimensions of the embedding layers were 300. The model was trained for five epochs during approximately one hour, obtaining a Test accuracy of 87.59% and 0.550 for the test loss. See *Image 2*.

```
Epoch: 05 | Epoch Time: 654.10
Train Loss: 0.068 | Train Acc: 97.41%
Test Loss: 0.550 | Test. Acc: 87.59%
```

Image 2. Results obtained from the one-layer MLP

In general terms, the two-layer MLP model performed better than the one-layer MLP where word embedding was applied. However, as it can be noticed in *Image 3* (see *Image 3*), the test accuracy of both classifiers decreased gradually as the number of epochs increased, whereas the training loss increased. This indicates that the model is overfitted. In other words, it systematically and proportionally improves at fitting the data that it sees (training data) while declining in the precision of fitting the data that it does not see (testing data).

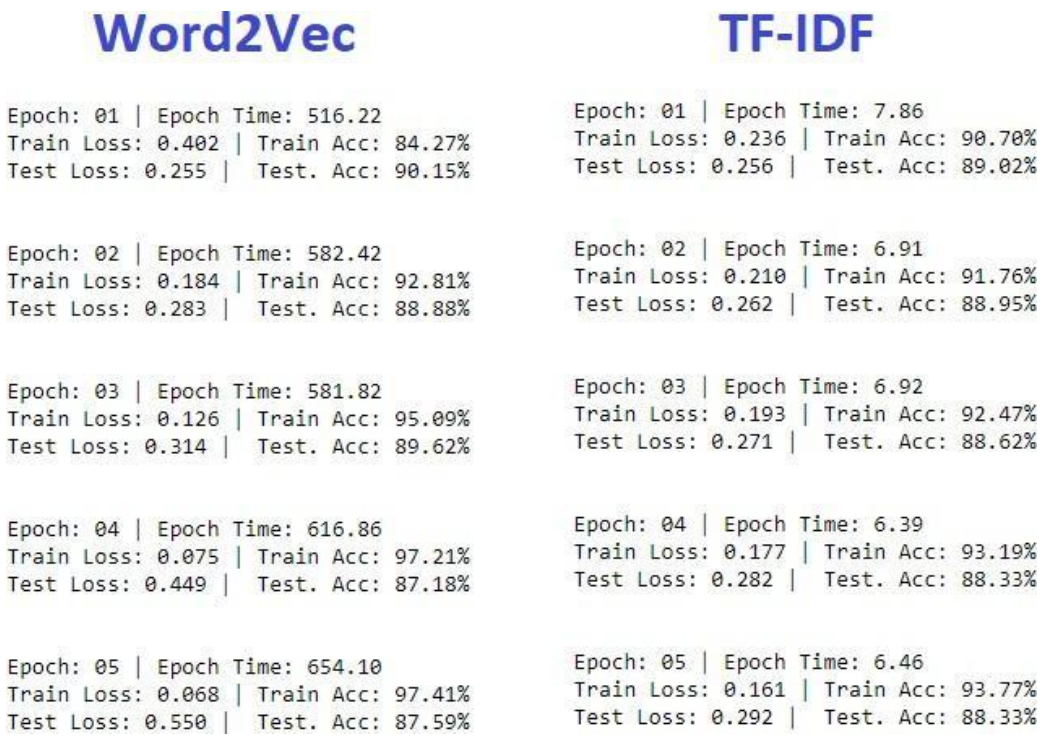

Image 3. Comparing the results obtained in the two MLP models

In this case, overfitting could be avoided by trying to decrease the complexity of the model. An example would be reducing the number of features, neurons or fully-connected layers in order to make the network smaller. A solution that could be applied to the one- layer MLP is to add a dropout layer. However, due to the complexity of the model, this implementation could not be applied successfully. The last solution to avoid overfitting is early stopping, that is to say, instead of training the model for a fixed number of epochs, it could be stopped as soon as the testing loss increases.

2. TRADITIONAL MODELS: LOGISTIC REGRESSION AND RANDOM FOREST CLASSIFIERS

The traditional methods selected were: Logistic Regression and Random Forest. Both of them depicted significant results. The Random Forest Classifier obtained a perfect train accuracy (1.0), whereas the test accuracy was set to 0.8543. On the other hand, the Logistic Regression classifier achieved 0.9138 accuracy, meanwhile the test accuracy represented nearly 0.891. See *Image 4.*

Random Forest		Logistic Regression	
Train Accuracy:	Test Accuracy:	Train Accuracy:	Test Accuracy:
1.0	0.8543	0.913825	0.891

Image 4. Comparing the results obtained in the two traditional ML models.

When comparing the test accuracy of the four models, it is noticed that the Logistic Regression model obtained the highest accuracy. On the contrary, the lowest test accuracy was generated by the Random Forest classifier.

3. CONCLUSION

To conclude, it is important to note that MLP has the characteristic of fully connected layers, which means that each perceptron is connected to every other perceptron. The problem is that the overall number of parameters might become quite large. Because of the redundancy in such large dimensions, sometimes this might be inefficient. Moreover, implementing neural networks is computationally expensive, compared to traditional models. Therefore, the implementation of the most suitable model depends on the task that has to be performed.