Deep Learning: Text Classification

Tanuj: SNR - 2038893 / ANR - 653883
Shreshtha Sharma: SNR - 2037590 / ANR - 434770

Master DSS
Group Assignment

Group 45
Oct 5, 2020

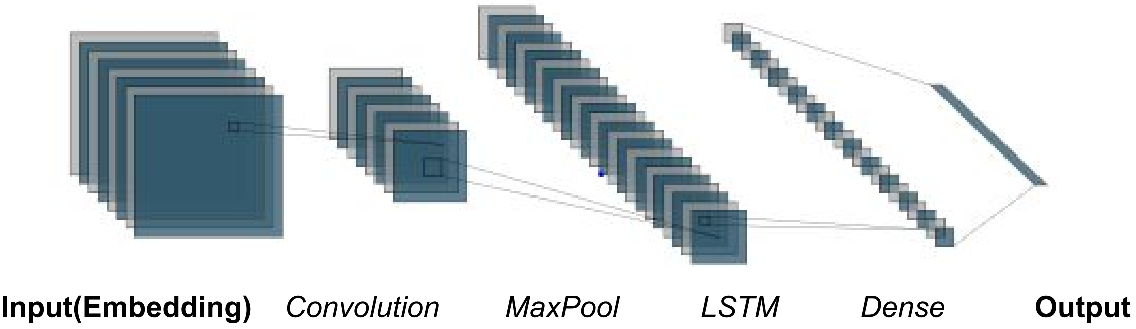
Tilburg University
Deep Learning

Data Preprocessing:

To start the project, data is loaded and prepared by means of data preprocessing. First, all features and labels were extracted from the dataset which consists of 188,775 text sentences with binary labels. Data is further tokenized and padded using Tokenizer before splitting it into train , validation and test splits using stratification sampling. Stratification is chosen to ensure that each label would be trained fairly, even when underrepresented.

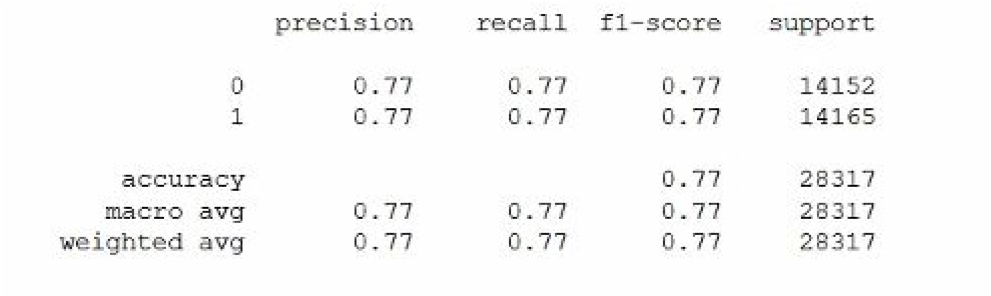
Neural Network Classification:

As a model, the neural network (CNN) is chosen. The model is trained using embedding, one convolution layer, max pooling and LSTM with Dense layer. The hyperparameters that have been tuned within the convolutional network are filters and kernel size .The epochs however were paired with an early stop function to prevent overfitting. As a result, the model often interrupted the prediction as it had already found an optimal value. The activation of the convolution layer is by “relu” function and padding is set to “Same”. For padding, “Valid” is used initially but this method proved to make the model less accurate and is computationally expensive. Further, “Sigmoid” is used for the Dense layer because it provides a confidence weight useful for binary classification.Within each layer hyperparameters have been tested to find what values result in an optimized accuracy on the validation set.The model is compiled with optimizer ‘’RMSprop’’ and loss is measured with ‘binary_crossentropy’. Finally, Accuracy is used as an evaluation metric for choosing the best performing model.



Result:

The best performing model is run with ten epochs, filter values of 128, a kernel size of 3, maxPooling layer of size 4 and 100 layers of LSTM are used. These are the default values as set in the attached python file “Group45” and can be reproduced by running the code. An early stop will interrupt the model after it has achieved the optimal accuracy. During training, an accuracy of around 77 percent is achieved for the training and validation set as shown in below table 1.1, further evaluating our model on the Actual Test data on Codalab, we achieved an accuracy of **77.6** percent on the blind test set with our account name **Group_45.**



```
              precision    recall  f1-score   support

           0       0.77      0.77      0.77     14152
           1       0.77      0.77      0.77     14165

    accuracy                           0.77     28317
   macro avg       0.77      0.77      0.77     28317
weighted avg       0.77      0.77      0.77     28317
```

Table 1 : Evaluation metric