In order to train a model, we need an appropriate dataset. For identifying the language of a document, several datasets exist such as Papluca, Tatoeba, WiLI and Language Detection (Kaggle). As Papluca [1] has balanced the train, validation and test sets with a considerable number of samples in each set, I chose this dataset to train the model for identifying the language of a text. The sentence samples are in one of the 20 languages and the size of the train, valid and test sets are 70,000 , 10,000 , 10,000 samples, respectively.

Firstly, I have trained the dense neural network model on the data. The features of each sentence are extracted based on the most important words in each studied language using TF-IDF [3]. The idea of the model is taken from [2] using feature engineering and deep neural network but it uses trigram most common characters as feature set whereas I used most important words. Also, I used more dense layers and used validation set during training as well to monitor the loss. Moreover, the parameters like number of features, epochs and batch_size are optimized and their best values based on the performance of the model were obtained.

Although the training accuracy of 93.72% is slightly higher than the test accuracy of 92.64% and since the validation loss starts to increase a bit while the training loss continues to decrease, it seems the model is overfitting. According to confusion matrix, majority of misclassification (around 407 test samples) belongs to Japanese language (ja) but misclassified with Chinese (zh). It seems it is hard for the model to distinguish between the words of these two language in spite of not having many common words (only 3 common most important words). Even feature extraction of most important bigram words is investigated but the accuracy dropped significantly. As the feature engineering is based on words, it causes the input feature set for the model to be sparse. Maybe this is the reason for the incapability of the model to classify the languages correctly.

However, the same dense neural network model is implemented with bigram character feature extraction; still the test accuracy (98.67%) is slightly lower than the training accuracy (99.66%), which shows that the model is overfitting. Nevertheless, the significant misclassification of Japanese sentences has been resolved, and all the languages can now be classified almost correctly.

The third implemented model is the Multinomial Naive Bayes Classifier (Mn-NB). To extract the features, again I used TF-IDF method to get the most important

characters and words in each language. The model was tried with the most important words as features of sentences but the train accuracy was around 74%. However, the performance of the Mn-NB model with 200 most important bigram characters was remarkably high, with a test accuracy of 98.93%, slightly higher than the training accuracy of 98.90%. Furthermore, similar to the dense neural network model with bigram characters of TF-IDF, the Mn-NB model did not significantly misclassify for any studied languages.

Additionally, At the end of each trained model, the model is tested on a Greek and Japanese new sentences and the predicted language is displayed.

All in all, the Mn-NB model with bigram character feature engineering on the Papluca dataset outperforms other implemented models with an insignificant number of misclassifications and no overfitting.

References:

[1] https://huggingface.co/datasets/papluca/language-identification

[2] https://towardsdatascience.com/deep-neural-network-language-identification-ae1c158f6a7d

[3] https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html