Carolina Raymond Rodrigues Student ID:210607000

ASSIGNMENT 1: SEQUENCE CLASSIFICATION.

Split the training data into 80% training, 20% development set.

The training data has been split into two subsets with help of the train_test_split function of the sklearn library. Moreover, the function's attribute random state has been set to 42, assuring that the content of each group remains the same during the classifier's optimization.

Once the data has been split, the provided code is re-run. The classification report outputs an F1 score of 54,57%. As expected, this value is lower than the one obtained before splitting the data, because the corpus is smaller, and less information is available for the learning process.

Error analysis 1: False positives.

According to the results of the previous section, the 5 classes with the lowest precision are *'B-Quote', 'B-Soundtrack', 'I-Opinion', 'I-Quote',* and *'I-Soundtrack'*.

A print method has been implemented to get a better insight of the errors. Words classified with one of those 5 classes will be compared with their ground truth labels. If they are not equal, we will have a false positive.

For most cases, the classifier was not able to predict correctly the biotag. For instance, in I-Quote, the common mistake was to tag as quote a plot's description, as it happened in the sentence *"What is the name of the fifth movie in the series about how you can't cheat death or fate".* Adding some context to the word might reduce these errors.

For I-opinion, tags corresponding to adjectives (i.e, classic, famous, funny) and verbs in present continuous (i.e, writing, directing) were not precise. Consequently, lemmatizing or stemming words may achieve better results when an inflectional form of a word is used.

Error analysis 2: False negatives.

The 5 classes with lowest recall are: *'B-Quote', 'B-Soundtrack', 'I-Opinion', 'I-Character_Name',* and *'I-Soundtrack'.*

To print the false negatives, a comparison of the words labeled with one of the 5 categories and the predictions made by the classifier will be done. In case of discrepancy, we will have an example of a false negative.

When false negatives were checked, something similar to the false positives happened. For instance, the sentence *"This Martin Scorsese film contains the classic moment of Robert DeNiro looking in the mirror and asking "You looking at me".* The word you is classified as plot when it is the beginning of the quote. Therefore, just as in the previous case, a little bit of context, lemmatisation, and stemming might help to reduce these errors.

In the case of I-Character_Name, the classifier was not able to identify correctly actors or the characters of a movie, for example in the sentence "What 2010 movie finds superhero alter ego Tony Stark once again battling evil and saving the day", Stark has been classified as part of the plot, when it is the leading role of the movie. Consequently, adding Name Entity Recognition (NER) may help to identify correctly the entities.

POS tags as features.

In this section, POS tags of the tokens are added as an attribute of the get_features. However, the function will only accept a list of strings. Thus, pre-processing is used to add the POS tag to the word. Inside the get_feature function, the token will be split into two and added to the features function as a word and a tag.

Adding the POS tags has increased to 56.25% the F1 score of the classifier. If we analyse this, we can see that in the cases with low recall or precision, the POS tags had a significant effect on the B-Quotes, almost doubling their values. However, in the case of the Soundtracks or the Characters' names, no changes are observed. Interestingly, in the Tony Stark example given in the previous section, Stark is tagged as a Proper Noun and yet, the classifier does not predict properly its label, which suggests that better algorithms must be added to classify correctly these words.

Feature experimentation and other optimization for optimal macro average.

An analysis of how the addition of different features affects the macro average F1 score of the classifier is done. The function that provides the best performance will be preserved and used to predict the biotags of the original test data.

Firstly, the optimization of the hyperparameters by adjusting the training options values was done. Three parameters were optimized: minimum frequency, c1, and c2. The former will help to restrict the vocabulary, eliminating frequent and infrequent terms. This value is 1 by default, therefore it was changed to 2 and 3. The results obtained favoured a minimum frequency of 2 since it significantly increased the F1 score to 59.44% while setting it to 3 resulted in 58%.

Additionally, c1 and c2 regularization parameters were added to avoid overfitting the model. Different values were tried; however, the best was obtained with c1=0.2 and c2=0.1.

The next step was feature engineering. Taking into account the results in the previous two sections, prefixes, stemming, lemmatization, NER, and some context was added to improve the performance.

The feature function included the extraction of suffixes up to a length of 3. Consequently, the same was tried but with prefixes, with the hope of improving the predictions of the adjectives and verbs that were wrongly tagged. This addition increased the overall F1 score to 61.7%, which suggested following a line where the root of the words should be extracted to improve the classification task.

Following this suggestion, the words were stemmed, the core-meaning bearing units were saved and added as features. Porter's Stemming algorithm was used in this part. However, although the F1 score rose, the difference was not significant (61.82%). While analysing the outputs, it was noticed that the main disadvantage of stemming is that the morphological variants that were obtained, were no real words (Beauty → beauti, picture →pictur…)

Since lemmatization considers the context while obtaining the root of the words, that was the next thing to try. Two different lexical databases were tried: Wordnet and TextBlop. The former one achieved better results, hitting 62%. Nevertheless, errors were still found. For instance, in the constituent *"The horrifying events…",* horrifying was not lemmatized as horrify. The reason behind this resides in the POS tag. The word was tagged as a noun instead of a verb, leading to a wrong lemmatization. Hence, adding some context may increase the performance.

Consequently, one window of the next and the previous token were included as features. In the case of being the first or last word, the assigned feature was CONTEXT_</s> and POSTCONTEXT_<\s>. The consequences of adding context were highly notable, the performance rocketed to 65.18%. the evaluation carried out, proved how the meaning of the words required some references. Nevertheless, adding 2 size windows was not optimal. The performance decreased to 64.9%, probably because the classifier does not have enough data to learn about.

Moreover, the words at the beginning and end of the sentence were rewarded with an extra feature BOS and EOS respectively. This attribute elevated the performance up to 65.64%.

Finally, following the analysis of sections 3, and 4. NER was performed using the Spacy framework. However, this reduced the overall F1 score to 65%. The disadvantage of this framework is that it used capitalization as one of the cues of identification. Since the words were not capitalized the algorithm was not able to work correctly. Therefore, the NER feature was discarded. In future works, it might be interesting to use the Bert framework, as some publications suggest.

After the feature optimization, the next step is to check how the model works on the original test data. The overall score was 62.13%. this value is lower than the one obtained in the held-out data as expected. The results suggest a good optimization and no overfitting of the parameters.