Carolina Raymond Rodrigues. Student ID: 21060700

ASSIGNMENT 2: DISTRIBUTIONAL SEMANTICS BASED EASTENDERS CHARACTER CLASSIFICATION.

The purpose of this assignment is to use pre-processing, feature extraction and transformation techniques to improve sixteen Eastenders characters vector representations. Improvements are assessed using a mean rank metric (test character documents that are closest to their corresponding training character documents), with 1 being the best feasible estimate. At the beginning of this assignment, the acquired mean rank value is 4.5. The techniques that have been applied in order to improve this metric will be explain in the following lines.

Q1. Improve pre-processing.

The pre-processing methods that have been used can be observed in the pre-processing function. Because a simple tokenisation on white space was performed within the function, a new complex technique was implemented using the NLTK Tokenizer Package. Once tokens were obtained, they were lowercased and punctuation marks were removed, resulting in a metric decrease of 4.375. Nevertheless, since Eastender is a soap opera that follows the stories of local inhabitants' daily lives, and Twitter is a social network in which users write about their lives, the decision to apply the TweetTokeniser function of the NLTK Tokenizer Package was made. The results revealed an improvement, the mean rank drop to 4.125. Therefore, from now on, character lines will be tokenised using TweetTokenizer function.

Consecutively, tokens are merged into a Bag of Words used to construct a vector for each of the Eastenders characters. Raising inter-variance among each character will result in a lower mean rank measure, therefore our purpose is to create unique vector representations of each character in order to make them more distinguishable.

To achieve our goal, the first step was to eliminate low-level vector information. Removing common English stop words such as "the" and "in" produced a significant improvement, the mean rank value decreased to 2.5 since more focus was given to important tokens.

Additionally, continuing with the idea of unique vector representation, lemmatisation was employed to maintain only the root of the tokens. WordNet, TextBlob and Spacy were all used as lexical databases. The first two reduced the metric's value to 2.19. Nonetheless, the later one achieved better results as the mean rank decreased to 2.06. Nevertheless, a deeper examination of the vector revealed that certain tokens were misspelled (faithful was written as faithful in some occasions), having a negative impact on the lemmatisation process. Therefore, the NLTK Autocorrection Package was used to emend the problem. As a result, the mean rank fell to 2.00.

Furthermore, a maximum token count frequency was set in order to eliminate frequent terms and consequently limiting the lexicon. Various frequency values were attempted, however, a maximum count of 58 generated the best possible outcome, mean rank was reduced to 1.875. On the other hand, establishing a minimum word frequency was not productive at all.

Q2. Improve linguistic feature.

Linguistic features such as POS-tags, N-grams and sentiment analysis were added to this part in order to increase distinguishability among vectors. Nonetheless, these strategies did not result in any improvements. Therefore, the decision of going back to the previous question was made. Some pre-processing approaches were altered to test whether better results were obtained for the current and next questions. Interestingly, if all the previous configurations were kept, but lowercase was removed, lemmatisation was done with WordNet and the maximum frequency of words was reduced to 51, the achieved mean rank value was 1.75. This result proves that even though a method doesn't seem appropriate at first (WordNet), its combination with other techniques may lead to a better performance.

After changes were applied, each one of the lines of a character were analysed and classified according to the sentiment it expressed. VARDER and TextBlob tools were used, sentences were

classified as positive, negative or neutral and then added as counts to the character vector representation. In both cases, just adding counts of positive and negative lines resulted in a mean rank of 1.69. however, including counts of neutral lines using the VARDER tool, benefited the mean rank since the value fell to 1.625.

Moreover, adding context may help to extract the aspects of each line, leading to vector uniqueness. Context was added in the form of N-gram counts. The consequences of including bigrams were highly noticeable, since the mean rank plummeted to 1.5. Nevertheless, using trigrams or higher order n-grams decreased the inter-variance among vector, mean rank rose to 1.94. We believe that the main reason of this retrogression relied on the fact that no additional or determining information was generated, the counts for most n-grams was 1.

On top of that, word such as “run” have different meanings depending on context. Adding a feature that combined the token with its corresponding POS-tag seemed a good way of dealing with the problem, however, it led to an increase on the mean rank (1.625), since the generated counts were similar for each character, confusing the classifier.

Finally, the given data was used to build a gender classifier. Different model classifier were applied (Gaussian, LGBM, Logistic Regression, SVC, NuSVC, LinearSVC, K-Neighbours and Decision Trees). Nevertheless, best accuracy was reached with Multinomial Naïve Bayes classifier (0.57), consequently it was the selected method. Nonetheless, adding gender a feature did not provide any improvements, the mean dramatically increased to 4.31, being therefore excluded as a feature. Better results may be achieved if the classifier accuracy was higher.

Q3. Add dialogue context data and features.

Previous and next lines spoken by another character in the same scene are included. Each sentence was “cleaned” in the same way as it was done in pre-processing (no punctuation, correct spelling, stop words removal and lemmatization). Next, tokens were added to the vector as a Bag of Words with the prefix PRE_ or AFT_ for each case respectively. In the case of the previous lines, just adding the words was not enough, a maximum frequency of 66 token counts was established, leading to a mean rank of 1.44. Furthermore, in the case of the next lines spoken by another character, a maximum frequency of 42 was also needed for the counts. This feature improved the performance noticeable, since the value plummeted to 1.125.

Nevertheless, adding more dialogue features such as scene information, episodes, scenes did not help to enhance the metric, therefore they were discarded as features.

Q4. Improve the vectorization method.

Until now, a Bag of Words has been used. However, in this section, TF-IDF technique is applied. This method represents how relevant each word is in a vector. Test vectors with similar relevant words to the training set will be then more precisely classified.

To implement this technique, a TF-IDF matrix transformer was added. The count vectorizer that produce sparse vectors from feature dicts, will be fitted to a TF-IDF matrix, and when needed will learn the idf vector using the fit method.

Different parameter values were tried, the most successful was the one in which L2 norm was used, the inverse document frequency wasn’t reweighted, and Laplace smoothing was added to the word frequencies. However, no improvements were made with respect to the previous question, not even in the mean cosine similarity (0.871) or accuracy (0.875).Even so, although it doesn’t work in this specific model, it might do in others, therefore, this technique was kept.

Q5. Select and test best vector representation method.

In this section we have to check how the model works on the original test data. The overall achieved mean rank value is 1.4375. 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.