Effect of different combinations of preprocessing methods for sentiment analysis through simple and complicated models

Gergo Gyori, Constantin-Bogdan Craciun, Jacob Andreas Sneding Rohde, Nicki Andersen
Data Science, IT University of Copenhagen
{gegy, cocr, jacro, nican}@itu.dk

Abstract

This document contains a project proposal for blabla bla

1 Introduction

Nowdays, the majority of the text in online product reviews, which is written by humans, is not written in an academic level (e.g.: grammatically incorrect, contains typos and non-interpretive language, emojis, etc), therefore the quality is varying. Through different preprocessing methods we investigate how the accuracy changes with the different combinations of preprocessing methods.

In this paper we seek to investigate how different combinations of preprocessing methods affect the accuracy of sentiment analysis with a simple RNN and more complicated models (Hugging Face transformers). We are especially interested in finding out if a simple RNN model with heavy preprocessing can outperform a superior model without heavy preprocessing.

2 Related Work

By trying different combinations of stopwords, lemmatizer and emoji recognition Mulki et. al. (2017) found that using stopwords (combination of Terrir and Snowball stopwords), the Treetagger lemmatizer and emoji recognition increased the accuracy with 3.4% as compared to only using stopwords, on a dataset of tweets. The authors found that the emojis had a good impact on their models. The reviews of for the present study don't include emojis, and one of the tasks is to find out whether further preprocessing can lead to a comparable increase in accuracy when lacking such emotion-carrying information as emojis. Camacho-Collados and Pilehvar (2017) also examined the effect on preprocessing methods for text classification. They found a statistically significant 0.7% reduction in accuracy on polarity detection on reviews from IMDb when using a lemmatizer from when not using a lemmatizer. This seems to be an opposite result from Mulki. et. al., even though their methodologies might make it hard to compare their results

3 Experiment Setup

3.1 Preprocessing methods

- Expansion of contractions
  Multiple words that have been contracted into one such as with "i'm" and "you're" are expanded into multiple words so "i'm" becomes "i am",
- Basic preprocessing
  Process when each word or meaningful characters (general punctuation's) are splitted. Through the method non-word instances (like "/n" or "[][]") are deleted.
- Grammar correction
  Mistyped words replaced with the closest match of a meaningful word. Note: Accuracy is not checked.
- Simplification of negations
  When an adjective is preceded by the word "not", "not" will be removed and the adjective will be replaced by its antonym. For example "not fast" becomes "slow".
- Lemmatization
  All words will be reduced to their base form in order to reduce the vocabulary size. For example "Walks, walking, walked" will all be changed to "walk", as they all have the same meaning.
- Removal of stop words

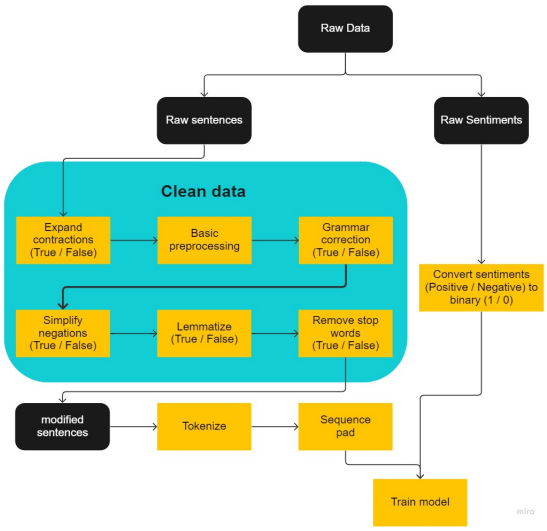

Figure 1: Pipeline of the process

3.2 Models

- RNN
- Hugging Face BERT (Bidirectional Encoder Representations) transformer. Bert is developed by employers at Google and got the best accuracies for some NLP tasks in 2018. is trained on the english Wikipedia corpus consisting of 2.5 billion words and BooksCorpus consisting of 800 million words. We use the standard version available from the Hugging Face repository as it comes out of the box with no fine tuning or post-training on our own data.

3.3 Methods

3.4 Setup

Through the training used we used 1000-5000-30000 sentences. Above training with 5000 sentences the train and test accuracy become steady, compared to use 30K sentence the overall accuracy was 1% lower.

3.5 RNN

Through the different train sizes the results could vary. Using 5000 sentences for training generated decent results for comparing the different methods

- Heavy emphasis on preprocessing, with minimal work on the model.
- Comparison of different training data instead of models
- Describing pipeline

4 Results

Different combinations of preprocessing methods led to different vocabulary size. In case of training with 5k sentences this is varying between 20k and 27k

Test Acc	BP	EC	GC	SN	Lmtz	StWrd	Sent.Len	Train Acc	Trainloss	Testloss
0.431673	1	0	1	0	0	1	40	0.236	11.784663	8.766422
0.641757	1	1	0	1	0	0	40	0.875	0.279350	0.660125
0.431673	1	1	1	1	0	1	40	0.236	11.784690	8.766422
0.568127	1	1	1	0	0	1	40	0.762	0.537670	0.805082
0.577031	1	1	0	1	0	1	40	0.769	0.530650	0.799561
0.578231	1	0	1	1	0	1	40	0.793	0.488759	0.763315
0.580332	1	0	0	1	0	1	40	0.796	0.505128	0.777870
0.586034	1	1	0	0	0	1	40	0.775	0.530483	0.825629
0.587035	1	0	0	0	0	1	40	0.795	0.506231	0.773721
0.601541	1	0	1	0	0	0	40	0.807	0.482596	0.745753
0.605242	1	1	0	0	0	0	40	0.804	0.485812	0.742799
0.609344	1	1	1	0	0	0	40	0.792	0.504578	0.723900
0.611845	1	1	1	1	0	0	40	0.821	0.437696	0.734119
0.619048	1	0	0	1	0	0	40	0.788	0.516308	0.713737
0.705082	1	0	1	1	0	0	40	0.931	0.176262	0.636511
0.764106	1	0	0	0	0	0	40	0.961	0.119621	0.680420

Figure 2: Different model's results

5 Discussion and Conclusion

The project is available on the below link:

https://github.com/csipapicsa/2ndYearProject-NLP

6 Further work

(The following are things we hope to add to the project before hand in next friday - a part from finishing what we already set out to do)

- Tf-IDf
- Subword tokenization
- Run the entire experiment on more datasets
- Use MLFlow to track experiments
- Synonym substitution

7 Electronically-available resources

The project is available on the below link:

https://github.com/csipapicsa/2ndYearProject-NLP

8 Referencies