Joke detection with text classification algorithms

Miriam Amin

WS 2019/20

1 Introduction

Humor is a fundamental property of humans. Although scholars are analyzing and studying humor since the Ancient Times, until today it is not understood completely. In contrast to other NLP-related problems, the computational treatment of humor is far behind.

Former research in computational humor was mainly carried out on humor generation and humor detection. As I showed in earlier work (Amin, 2019), none of the humor generators presented so far were able to produce human-like humor. From my investigations I concluded two approaches which seemed promising for the advancement of joke generators – a generative and a restrictive approach. A generative approach to humor generation would aim at exclusively producing humorous output by preselecting suitable topics to joke about. A restrictive approach on the other hand would consist of two systems: A system that produces texts with structural features of jokes and a second humor detection system that works as a filter letting only the humorous texts pass. One approach for such a filter would be a neural network for text classification with the target classes `joke` and `no joke`.

The aim of this project is to assess the feasability of current algorithms for text classification for the application as such a joke detector. In the following I will briefly present related work and earlier systems for joke detection. I will proceed by outlining the intended method and the data set that will be used for training the classifiers.

2 Related Work

Several attempts have been made to classify different forms of humor. Chen and Soo (2018) discriminate English and Chinese one-liners and puns from news text with similar vocabulary. de Oliveira and Rodrigo (2015) detect humor in Yelp reviews. Both groups of authors found that Convolutional Neural Networks (CNN) yielded the best results for accuracy.

3 Methods

For the joke classification task, I trained a logistic regression classifier and a multilayer perceptron with a dataset of 1106 documents labeled as jokes and 1106 documents labeled as non-jokes, yiedling a sample size of 2212 documents in total. In the following, I describe the processes of data gathering and classifier training in greater detail.

3.1 Data Compilation

The training dataset consists of labeled jokes and non-jokes. The positive jokes examples originate from different websites, crawled by Pungas (2017) and Moudgil (2017). This dataset contains 28.471 jokes. Compiling a negative data set is more challenging. The documents have to be linguistically similar to jokes, but should not be funny.

I identified the following criteria for the documents in order to qualify as non-joke sample for the classification task:

1. Narrative/fictional style
2. Length (number of sentences) of a typical joke
3. Language similar to jokes

In order to satisfy the stilistic requirements, I used a dataset of all movie plots of Wikipedia found on Kaggle (Revanth, 2018). This dataset contains approximately 35.000 documents. Using this dataset as a base, I applied several methods to compile the data in order to comply with requirement 2) and 3).

Jokes come in a variety of shapes and lengths. While a question and answer joke consists of two sentences typically, a longer narrative joke can span over the length of a complete paragraph. Graph 1 shows the distribution of joke lengths in the corpus.

The vast majority of jokes have a length of two sentences, which may result from a large number of question-answer jokes in the corpus. The mean joke length in the corpus is seven sentences. For the subsequent investigataions, I considered only texts with a length of six sentences. I chose this number because it is close the the mean number of sentences over the complete corpus, but has a slightly higher absolute frequency than jokes with seven sentences.

Since each of the movie plots in the dataset consists of more sentences than an average joke, I decided to split the entire movie plot corpus into overlapping sentence-6-grams. This procedure renders 80.897 nonjoke candiate documents. At the same time, I extracted only jokes with a length of 6 sentences from the original joke collection, yielding 1106 jokes. This method of only considering documents with a fixed length decreases the complexity of the task of compiling a negative joke dataset to a manageable level.

In a last step, I found the most similiar nonjoke for each of the jokes in the reduced joke corpus. For this purpose, I calculated a similarity matrix conveying the cosine similarity between each joke and all nonjoke candidates. The cosine similarity is computed on the basis of the docments' word embeddings trained from a word2vec model of the

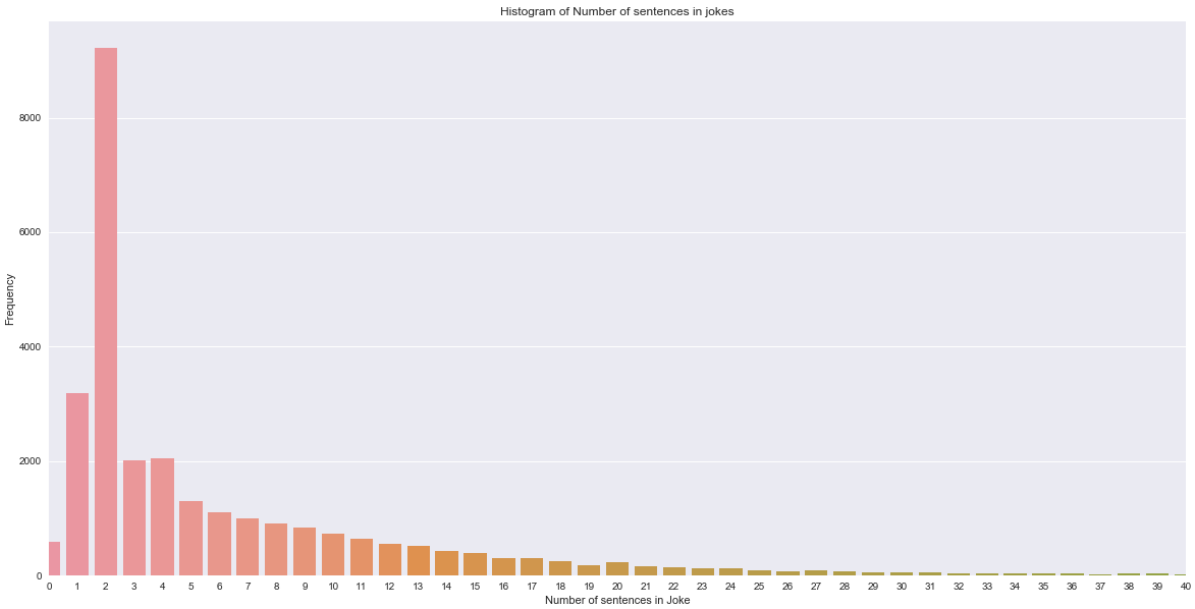

Figure 1: Distribution of joke lengths in sentences

spaCy python library (Honnibal and Montani, 2017). For each joke, the nonjoke with the highest similarity is selected. The collection of all nonjokes serves as the corpus of negative examples for the training of the classifier.

3.2 Classifier training

For the purpose of joke classfication, I tested two different classification models:

1. Logistic regression

2. Multilayer Perceptron

For the logistic regression model, I created the feature vectors by vectorizing the tokens with CountVectorizer from the scikitlearn Python package. CountVectorizer replaces the document by a vocabulary-sized vector where each value of the vector represents the frequency of the respective vocabulary item in the document. Punctuation and other special characters are removed. After training, the classifier yields a mean accuracy of 99.2 % on the given test data and labels.

The second model is a simple feedforward multilayer perceptron. It consists of two densely conntected neural network layers. This model reuses the count-vectorized dataset from the logistic model. After training the model for five epochs, training and validation accuracy are satisfactory. The model reaches 99.8 % testing accuracy with a relativaley low loss. The validation loss curve in Graph 2 implies that further training would lead to an increase and consequently to overfitting.

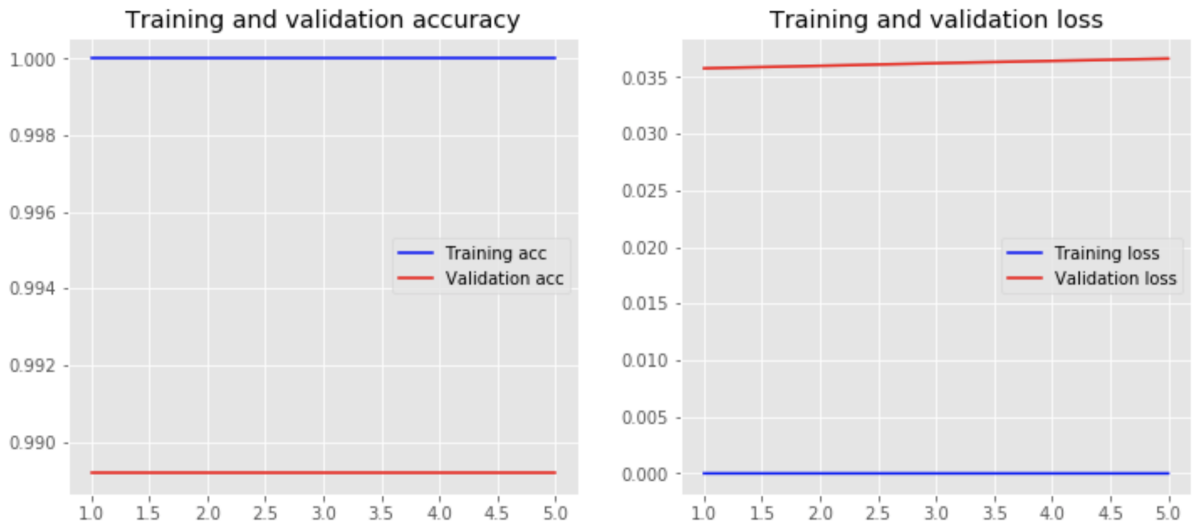

Figure 2: Multilayer Perceptron learning curves

	Logistic Regression	Multilevel Perceptron
Joke Dataset	0.976	0.986
Movie Plot Dataset	0.989	0.988
Mean	0.982	0,987

Table 1: Accuracies of classifiers on different datasets

4 Results

In order to test the generalizability of the models, they will be tested on the entire initial joke and non-joke data, before length-filtering and similarity search. Table 1 shows the accuracy of the two models on the joke and movie data as well as the mean over both datasets.

To test the models' general ability to capture humor, both models are applied on two science fiction novels: Douglas Adams's The Hitchhiker's Guide to the Galaxy and Salman Rushdie's Satanic Verses. For that purpose, the books will be split into sentence-6-grams that will be classified as joke or no joke. Then, the percentage of humor in the books is calculated. Douglas Adams' work is hereby considered as humorous whereas Rushdies novel is considered as non-humorous. This human estimation should be reflected by the algorithms. Table 2 shows the predictions of the models on these two books.

5 Discussion

First of all, the level of accuracy on the training process and while testing and on the entire dataset is satisfactory. It is surprising that these two relatively simple models both

	Logistic Regression	Multilevel Perceptron
Douglas Adams	0.940	0.975
Salman Rushdie	0.767	0.893

Table 2: Level of humorousness according to the two models in different novels

reached accuracys of far over 90 %. I initially planned on implementing an convoluational neural network with word embeddings as feature vectors. However, after yielding these convincing results with simpler means, the implementation of a more complex architecture did not seam necessary any more.

Nevertheless, the generalizability of these models is questionable. Applying the algorithms on out-of-domain novels, their classification of the content does not necessaryly meet the human estimation. Obviously, there is no objective humor score for the both novels. However, since Rushdies novel is not humorous, but dramatic, the model's humorousness scores are way too high. Though the logistic regression classifier assigns a slightly lower percentage of humorous content to Rushdies work, it is still relativaley high. This might be due to the fact that Rushdie uses a language that is often used in jokes. However, a joke detection algorithm should be able to detect humor and jokes independentely from the vocabulary.

The reason for these missclassifications are most probably due to the simpleness of the vectorization of the docuemnts' text. A simple CountVectorizer as used here classifies texts merely by the use of vocabulary. More advanced pretrained word embeddings, such as word2vec or GloVe can already take a word's context into consideration. However, humor is created by complex semantic mechanisms that frequently invoke ambiguity. To capture ambiguity, state-of-the-art deep language models such as BERT or ELMo can be applied. Also, a larger training dataset would be needed to provide enough samples to allow the generalizability of humor.

References

Amin, M. (2019). *Computational Humor - Automatic Generation of Jokes*. Bachelor Thesis, Leipzig University.

Chen, P.-Y. and Soo, V.-W. (2018). Humor Recognition Using Deep Learning. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, volume 2 of *Short Paper*, pages 113–117.

de Oliveira, L. and Rodrigo, A. L. (2015). Humor Detection in Yelp reviews.

Honnibal, M. and Montani, I. (2017). spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing.

Moudgil, A. (2017). Python scripts for building 'Short Jokes' dataset, featured on Kaggle. *GitHub repository.*

Pungas, T. (2017). A dataset of English plaintext jokes. *GitHub repository.* https://github.com/taivop/joke-dataset.

Revanth, G. (2018). Stories of American movies | Kaggle. https://www.kaggle.com/revanth9/wikipediaplotdata.