Fictional Character Classification

Alexandros Chimonas
i6151603

Simon Osadchii
i6166701

Abstract

In this paper, authors attempt to describe approaches to Profile classification of fictional characters using only their sentences that were taken from a script. Different Natural Language Processing (NLP) techniques and machine learning classifiers have been used in order to compare techniques.

1 Introduction

A lot of research in the field of Natural Language Processing(NLP) is done recently. Topic identification and sentiment analysis, as well as text classification found its' are widely implemented and used. However, not that much research regarding speech classification is being done. This paper aims to find an optimal strategy using known NLP and machine learning techniques to assess it performance on classifying fictional characters by text lines from a script where these characters are present. For this purpose two data sets will be used, namely the scripts for all Simpsons Family episodes for the first 27 seasons and for all South Park episodes for the first 18 seasons (**Apendix A. Datasets**). The aim of this paper is to find the algorithm that gets the best accuracy in this task. Namely the algorithms used will be simple classifiers like Naive-Bayes and Logistic Regression. Then two different neural network architectures will be used, specifically Convolutional Neural Networks (CNN) and Long-term Short-Term Memory (LSTM). The language that is processed is English. Provided in the paper approach can be then used in e-security. Classifiers can be trained to detect if user has been hacked by assessing how likely the message that was sent by some user corresponds to the linguistic profile of user and hence, prevent identity theft with the help of NLP.

2 Pre-Processing

The two data sets were used to get the sentences that were said from each character. Due to the fact that the Simpsons data set contained more seasons, therefore more episodes and more lines per character, the data set was heavily unbalanced. For the purposes of balancing the data, we have tested a classifier (Section 3.1) with an unbalanced data set as showed in Figure 1 and on a balanced data set. The results favor the balanced data therefore we have used that table. The data was balanced using imblearn (imbalanced learning) package in python, which over sampled the data using the Synthetic Minority Over-sampling Technique (SMOTE) in order to have a fully balanced data set.

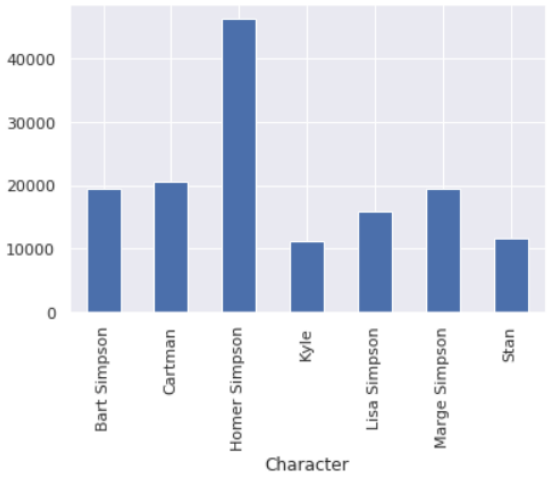

Figure 1: Amount of sentences per character

Furthermore, nltk python packages process the text files. More specifically, sentence and word tokenization was used. Stop-words (predefined set of words in English that have little to no meaning) were removed from the text, the lemmatizer and stemmer were also used to clean

	Precision	Recall	F1-score
Homer Simpson	0.40	0.86	0.55
Marge Simpson	0.50	0.21	0.30
Bart Simpson	0.41	0.17	0.24
Lisa Simpson	0.40	0.16	0.23
Cartman	0.50	0.40	0.44
Kyle	0.51	0.13	0.20
Stan	0.43	0.13	0.20
Micro avg	0.42	0.42	0.42
Macro avg	0.45	0.29	0.31
Weighted avg	0.44	0.42	0.37

Table 1: Results with Naive-Bayes classifier on the unbalanced data set

	Precision	Recall	F1-score
Homer Simpson	0.51	0.40	0.45
Marge Simpson	0.34	0.41	0.37
Bart Simpson	0.28	0.27	0.28
Lisa Simpson	0.30	0.29	0.29
Cartman	0.41	0.41	0.41
Kyle	0.23	0.24	0.24
Stan	0.21	0.34	0.26
Micro avg	0.36	0.36	0.36
Macro avg	0.33	0.34	0.33
Weighted avg	0.37	0.36	0.36

Table 2: Results with Naive-Bayes classifier on the balanced data set

Layer	**Parameters**	**Output Shap**
Embedding		(25,25)
Convolutional	filters = 128 and kernel size = 5	(21,128)
Max Pooling		(128)
Dense	hidden units = 25	(25)
Dense	hidden units = 7	(7)

Table 3: Architecture of Convolutional Neural Network

the text. Moreover, we also make use of some simple regular expressions, to reduce noisy data and some slang expressions

3 Classification

In the following section different classification methods are showed that were used, along with their performance. The subsections will be split by each classifier and after briefly explaining how the classifier works and its architecture, its results will be shown. There are 7 classes that each model is trying to learn and map to the text input. Each class represents a character from the cartoon series. These characters are : Homer Simpson, Marge Simpson, Lisa Simpson and Bart Simpson from 'Simpsons family' and Cartman, Kyle and Stan from 'South Park". Decision to take only those characters from the whole data set is explained by the number of sentences spoken by them. most frequent narrators were chosen for further analysis. Figure 1 shows the number of sentences for each character that was chosen.

3.1 Naive Bayes

The first classifier that was tested is the Naive Bayes classifier. Naive Bayes is a probabilistic classifier model. Naive Bayes has been trained twice, on two different data sets. The first data set was the unbalanced data set and the second one is on a fully balanced data set after using oversampling on the data. The vocabulary is represented by the Bag of Words (BoW) approach.

It is important to see how the classifier performs in these two data sets. Due to the nature of this project, it is more valuable to have more balanced recall metric. After balancing the data it is obvious that Homer Simpson's recall metric has dropped dramatically, but the other classes have increased, giving more balanced results that do not favour any class in particular. For this reason, the rest of the classifiers will use the balanced data set.

3.2 Convolutional Neural Networks

We have used a simple Convolutional Neural Network (CNN)(Kim, 2014) architecture for the same task. We have implemented a very simple network that takes as input the bag of words representation. The architecture of the CNN is done by first having a trainable Embedding Layer, followed by a Convolutional Layer, a Max Pooling Layer and finally two Dense Layers to finalize the classification.

3.3 Recurrent Neural Networks

For the Recurrent Neural Network we use a simple architecture of Long-Term Short-Term Memory (LSTM)(Le-Hong and Le, 2018) model. As previous, the model starts with a trainable Embedding Layer. Followed by an LSTM layer and a

Layer	**Parameters**	**Output Shape**
Embedding		(25,25)
LSTM	units = 56	(56)
Dense	hidden units = 7	(7)

Table 4: Architecture of LSTM Neural Network

Dense layer in the end to finally classify.

4 Results

After different approaches on sentence classification the following models have achieved the results shown in Figure 2. The neural networks as expected perform better than the other models. There is still room for improvement in the neural network models as only simple architectures have been used, with not a lot of parameter tuning. Moreover,

Algorithm	Accuracy
Logistic Regression	37%
Naive Bayes	38.5%
CNN	41.23%
CNN	41.23%
LASTM	43%

Figure 2: Performance per algorithm

5 Conclusions and Future Work

From the results, one can argue that the classification accuracy and recall are not impressive. However, the complexity of the initial problem should be taken into account. Each character used both negative and positive sentences on different topics, such as politics, sports, society etc. It is important to note, that the goal of the algorithms were just to see if current NLP techniques can differentiate and predict which sentence is said by which character. Algorithms and techniques used in the paper showed that it is possible to non-randomly distinguish between fictional characters. Furthermore, some can claim that the non-neural network classifiers are better, as they give similar results, but require less training time than neural networks. It's important to draw attention to the fact that due to inefficient computation power and time, the architecture and hyper-parameters of the neural networks could not be optimized to give the best results. Furthermore, the pre-processing could be improved with more sophisticated text cleaning methods, name-entity recognition of the input text and deeper parameter-tuning.

References

Yoon Kim. 2014. Convolutional neural networks for sentence classification. *CoRR*, abs/1408.5882.

Phuong Le-Hong and Anh-Cuong Le. 2018. A comparative study of neural network models for sentence classification. *CoRR*, abs/1810.01656.

A Appendices

A.1 Datasets

- South Park: https://www.kaggle.com/tovarischsukhov/southparklinesAllseasons.csv
- Simpsons Family: https://www.kaggle.com/pierremegret/dialogue-lines-of-the-simpsons