Speaker Identification Through Text Analysis

Final Project

DATS 6312 NLP for Data Science Project

Team 1: Brian, Amna, Saurav

Introduction

The aim of this project was to build a model, that given a sample line of dialog, could identify the character from a TV series most likely to have spoken it. The data set chosen for the project and published to Kaggle is 18 seasons worth of dialog from the animated Comedy Central series, South Park[1].

The selected data set offers 70,896 observations, with each observation being one or more sentences of dialog spoken by a character on the show. The data set attributes the dialog to 3,950 unique characters, but an overwhelming majority of them speak only a few lines (1-10 observations), while the main characters (Cartman, Stan, and Kyle) are associated with 7,248 to 9,843 observations each. To avoid imbalanced classes of data, our model limited its scope to analyzing the top three characters.

As detailed in this report, significant challenges were encountered in obtaining a suitable F-1 and accuracy score. Our initial plan of applying multiple NLP techniques in tandem and weighting their classifications together, was abandoned in favor of testing a number of classifiers on the data. Ultimately, it was found that the Support Vector Machine (SVM) classifier provided the best results, achieving a precision of .54, a recall of 0.80, an F-1 score of 0.64, and an accuracy score of 0.52 for the characters Cartman.

Background

Our review of previous work around this problem area focused primarily on Authorship Identification. Researchers, such as Liuyu Zhou and Huafie Wang of Stanford University, leveraged word vectors produced by the GloVe framework, in combination with Recurrent Neural Networks[2], ultimately producing a 0.6 F score. They conclude that the RNN approach does not offer a significant advantage over other machine learning methods like Random Forest Classifiers or Extra Trees Classifiers.

Another team from Stanford, Chen Qian, Tianchang He, and Rao Zhang, also applied advanced deep learning techniques described in their paper, “Deep Learning based Authorship Identification” [3]. They however chose to focus on both the sentence and article level embeddings, rather than at the word level. Their paper declares the article-based Gated Recurrent Unit approach performed the best, with their two datasets achieving an accuracy of 69.1% and 89.2%.

Scope

The initial plan for the proposed model was to leverage multiple techniques in parallel and arrive at the optimal classification by weighting the results of each classification together. The plan centered on Named

[1] https://www.kaggle.com/tovarischsukhov/southparklines
[2] https://cs224d.stanford.edu/reports/ZhouWang.pdf
[3] https://web.stanford.edu/class/archive/cs/cs224n/cs224n.1174/reports/2760185.pdf

Entity Resolution (NER), Term Frequency – Inverse Document Frequency (TF-IDF), and sentence embeddings.

The NER technique intended to analyze the sample line of dialog to identify any named reference to other characters. The occurrence of one or more character's names in the sample dialog would allow us to negatively weight or exclude them from consideration for classification, as English speakers usually do not refer to themselves in the third person. For example, if the sample line of dialog were "Leave me alone, Stan", then it could be assumed that the speaker is not Stan. Eliminating or discounting one or more speakers from consideration would increase the probability of identifying the correct speaker.

If each of the three TV characters were written with unique vocabularies over the seasons being analyzed, TF-IDF could aid in classifying a sample line of dialog based upon the individual words or n-grams contained within it. Each character's dialog compiled into separate documents and a document-term matrix created, TF-IDF analysis would allow for the term-based classifications.

The final technique of the initial plan was to utilize sentence embeddings. Encoding each sentence of dialog into a dense vector, then vectorizing the sample line of dialog would allow for classification of likeliest speaker by identifying those characters whose dialog was closest to the sample dialog in vector space.

After exploring each of the three techniques that comprised the original plan and assessing its feasibility, the scope was expanded to include traditional classifiers such as Naïve Bayes, SVM, KNN, and Random Forest, as well as a Recurrent Neural Network, using Long Short Term Memory architecture.

Outcome

Initial Plan

The Spacy package was chosen to implement the NER and sentence embeddings approaches that were part of the initial plan. Patterns for each of the character names were crafted using Spacy's entity ruler feature. The custom rule patterns, combined with the Spacy NER capability found in the small English model, successfully identified character names in dialog. Upon closer examination of those matches, it was discovered that significant lines of dialog in the show contain the speakers name, thus negating the effectiveness of negatively weighting or eliminating characters from consideration for classification. This phenomenon is somewhat unique to the way this show is written and is likely more viable for more formal English prose.

Spacy's sentence vectorizations using the medium English model, allowed for the implementation of sentence embedding. A vector space was created for each character based upon their dialog and a similarity score was calculated for each sample line of dialog across all characters. Testing of these scores by using sample lines of dialog unique to a given character and taken verbatim from their data, was expected to yield a near perfect similarity score for that character and lesser scores for the other characters. However, testing of this sentence embedding technique showed unexpected results, with each character showing near similarity scores and the actual speaker of the sample line of dialog never achieving the highest score. An example of this result is shown in the following screen snippet, with a sample line of dialog taken from the character Cartman and the similarity score assigned to him of 0.9210823 being the lowest score among all characters.

```
query_doc = nlp("Well, this was a great meeting. Wasn't it, guys? But, we gotta get goin' to schoo
l now.")

print(cartman_doc.similarity(query_doc))
print(stan_doc.similarity(query_doc))
print(kyle_doc.similarity(query_doc))
```

```
0.9210823907870881
0.929157707737528
0.9214379667051773
```

It may be that the characters' dialog is too similar across this corpus to effectively use it for sentence embeddings like this. The TV show's dialog could also differ significantly from the model upon which Spacy's medium English model was built, thus requiring custom training based upon the model for additional vectorizations.

TF-IDF with Stop Words

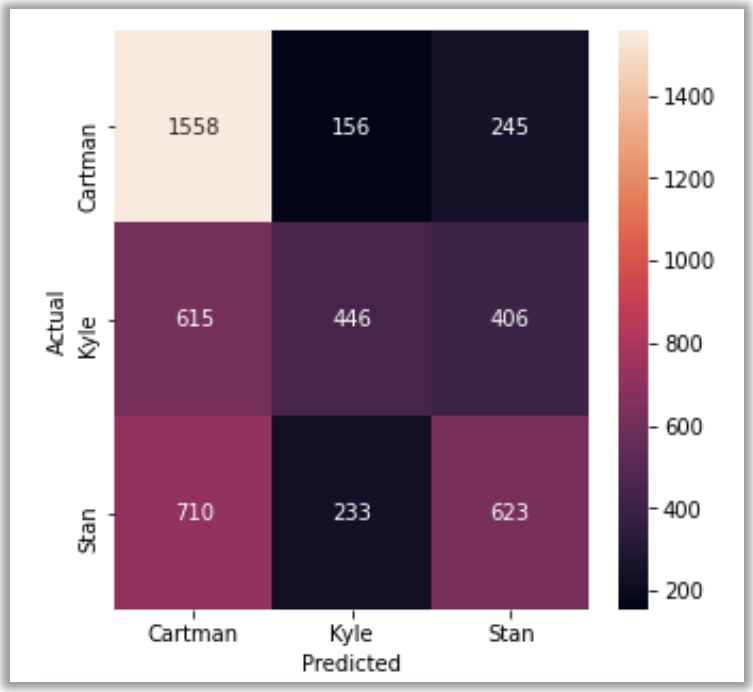
With two of the tree techniques of the initial plan showing unfavorable results, it was decided to apply more classical classification approaches. The third technique that we used for feature extraction and modeling was, TF-IDF, for rendering more importance to rarer words and to create a vector of these from each dialogue spoken by the character. N-gram range was set to (1,3) to include unigrams, bigrams, and trigrams, taking into account the order of words. Stop words were not removed.

As a result, 215,339 features were extracted from the data set which were passed to the following text classification models as input features for training and testing set:

Classifier	Character	Precision	Recall	F-1 score	Accuracy
Naïve Bayes	Cartman	0.48	0.90	0.62	0.49
	Kyle	0.60	0.15	0.24	
	Stan	0.51	0.31	0.38	
SVM	Cartman	0.54	0.80	0.64	0.52
	Kyle	0.53	0.30	0.39	
	Stan	0.49	0.40	0.44	
KNN	Cartman	0.44	0.84	0.58	0.43
	Kyle	0.42	0.15	0.22	
	Stan	0.45	0.20	0.28	
Random Forest	Cartman	0.56	0.64	0.60	0.49
	Kyle	0.46	0.32	0.37	
	Stan	0.42	0.47	0.44	

From the table above, the highest accuracy score was produced by SVM ~0.52% on test set whereas of all the models used, KNN (with k=3) gave the lowest accuracy score of ~0.43%. As an alternative to classification accuracy, we also take a closer look at F-1 scores (harmonic mean of precision and recall) which indicates higher F-1 score for character Cartman, followed by Stan and then Kyle in all models except KNN.

As expected from the accuracy scores, the confusion matrix below (of best performing model i.e. SVM) depicts that almost half of Cartman dialogues were confused with those of Stan. Similarly, more than half of Kyle's dialogues were confused with those of Stan's and also vice versa.



One of the possible reasons that SVM performed better than other models is its ability to learn (determine the decision boundary) independent of the dimensionality of feature space. For numerical data sets we usually try to avoid high dimensional feature space by assuming that most of the features are irrelevant and they try to remove them using feature selection. But text classification on the other hand, assumes that most of the features are relevant for information gain hence they must be retained. So SVM produced promising results for high dimensional input space problems like text classification.

TF-IDF without Stop Words

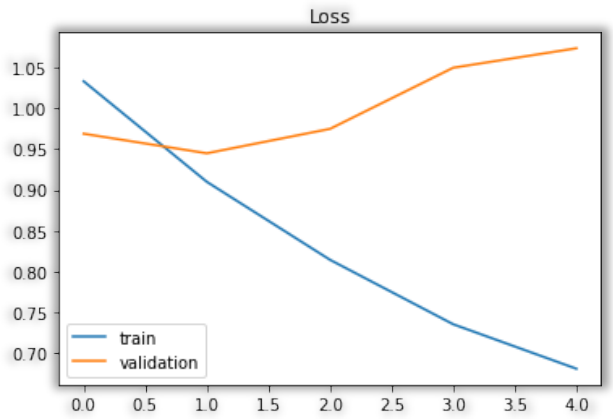
In an attempt to further simplify the textual data, stop words were removed before extracting features using TF-IDF. This time a total of 128,633 features were extracted.

Classifier	Character	Precision	Recall	F-1 score	Accuracy
Naïve Bayes	Cartman	0.49	0.87	0.63	0.49
	Kyle	0.57	0.19	0.29	
	Stan	0.49	0.32	0.38	
SVM	Cartman	0.53	0.77	0.63	0.51
	Kyle	0.49	0.31	0.38	
	Stan	0.48	0.38	0.42	
KNN	Cartman	0.45	0.17	0.25	0.33
	Kyle	0.38	0.12	0.18	
	Stan	0.31	0.76	0.44	
Random Forest	Cartman	0.59	0.58	0.59	0.48
	Kyle	0.44	0.36	0.40	
	Stan	0.42	0.50	0.46	

From comparative results displayed above, it was observed that although the number of features is roughly reduced by half, accuracy scores for SVM, Naïve Bayes and Random Forest are almost exactly the same. However, the performance of KNN has drastically aggravated from 43% to 33%. Possible reason for low performance of KNN (as compared to NB and RF that take into account the probabilities of likelihood and a hierarchical architecture respectively for prediction making) might be its simple design that takes into account just only the local approximations and then classifies new cases based on distance functions as a similarity measure. But we already saw when implementing sentence embedding that similarity measures of dialogues spoken by all characters (not just one) are pretty high (almost indistinguishable for machine) hence making it difficult for KNN to produce robust results.

Recurrent Neural Network (RNN)

Apart from the traditional classification techniques we also employed a simple Recurrent Neural Network using Long Short Term Memory architecture as it stores the information for current as well as neighboring features for prediction. From graph below we see that the loss is decreasing for training set but increasing for validation set with each epoch, indicating signs of overfitting. The final accuracy score is almost the same as that obtained from SVM.



RNN	Loss	Accuracy
Validation Set (Training)	0.97	0.52
Test Set	1	0.51

As mentioned in Background/Literature review section of this report, most of the techniques pertaining to text classification domain revolve around Neural Networks due to their potential to reach high accuracy with less need for feature engineering. But one important point to remember about deep learning algorithms, is that they require much more training data (i.e. at least in millions). In fact deep learning classifiers continue to get better, the more data is fed into them. In our case we had our training examples only in units of thousands so this is primary reason why RNN produced results that are only as good as SVM.

Challenges

The most significant challenge encountered on this project is fully interpreting and accounting for the peculiarities demonstrated by these tools and techniques. Many of these were new to the team members and required significant investments in learning to apply them. While there is significant documentation and examples online around the syntax for invoking the packages, much less is available on interpreting the output and adjusting the inputs for better results.

By reviewing the word clouds, employing different techniques (Sentence Embedding, TF-IDF) and results obtained (Accuracy score, confusion matrix), we concluded that the stylistic features of the three main character's dialog is largely indistinguishable from each other. A dataset with more variance in dialog could yield better results without refactoring the logic of the models.

Lastly, the problem space that we chose for this project originally belonged to the audio/visual domain using speech and image signals. We struggled to work in this problem space using only text based linguistic clues.

Future Work

Plans for future work fall into two general categories: refinement and exploration. The refinement category calls for continued refactoring of the existing scripts to better utilize the selected tools. Increased accuracy and improved F-1 scores are likely possible.

The exploration category encompasses the approaches described in the literature review and in researching the current tools. It appears that deep learning techniques and embeddings hold the best promise for significantly improved results. Implementation of these tools requires a yet undeveloped skillset for those of us on the project.