[[1]](#footnote-1)

Who's that Comedian: Predicting Comedians Using Markov Chains and Sentiment Analysis(December 2017)

Marina Schwadron, Jamie Catterall, Matana Maron

F

# INTRODUCTION

or our project we wanted to try and implement a program that could guess what comedian wrote an unidentified stand-up transcript. Based on previous literature and comedy analysis that has been done, we decided to create both a markov chain and naive bayes program. In this report we will discuss how exactly we implemented these two solutions and what challenges we had. We will also go into detail about the results, from which we can ultimately conclude that it is not effective to judge comedic stand-up transcripts based only on word order and frequency, as is possible with literary passages used in our past homework assignments.

# Problem

Recently, machine learning has been used to identify speech patterns in people's’ writings; this capability has made it possible to determine with reasonable certainty who has written what. This can be useful in many situations, but perhaps not in a comedic one. While machine learning has been used to analyze the patterns of natural speech and writing, there remains a gap between the capabilities of machines to learn to recognize natural speech patterns and the capabilities machines have to identify the nuances present in more subjective types of language, like humor. Analyzing the patterns of a comedian’s joke telling is a unique problem in terms of analyzing a person’s writing generally, since jokes use abstract elements of non-literal and nonsensical human understanding that make many jokes funny. For instance, take John Mulaney’s quick one-liner “I was once on the telephone with Blockbuster video, which is a very old-fashioned sentence.” This is funny not because of the syntax of the joke or the words used, but because of the human understanding that Blockbuster is outdated. This common understanding is what makes the joke funny, however a computer that has been trained on previous conversations might not know Blockbuster’s history, and so would not be able to register this line as being funny.

Jokes can be differentiated from conventional literature in that they do not follow a typical pattern; what makes them funny is the element of surprise, among other unique patterns of speech. Because of this distinction, we chose to try and analyze jokes more specifically to see who has written them, in a way that is distinct from the analysis of conventional literature. We did this analysis by implementing both a naive bayes and markov chain approach, in order to look at a combination of stylogenetics and conventional speech patterns used in stand-up comedy that has not yet been analyzed.

We used transcripts from nine different comedians as our data. We found the transcripts online, and chose comedians based on who had the most available material. Most comedians we used had transcripts from at least two different comedy specials available, each special being about an hour long, along with other supplemental, smaller transcripts that we may have found. The test passages are typically small passages lifted right out of the transcripts of the hour-long specials

We are referencing sources from two schools of study that people have used in the past: Stylogenetics and Computational Humor. Stylogenetics usually works with literature to construct a “genome” of an author based on characteristics of their style, then takes an input text and finds the figure with the most similar style. The process of identifying the stylistic features of each comedian whose monologues are analyzed fits under the category of stylogenetics. We reviewed two pre-existing stylogenetic studies. The first uses clustering-based analysis1, and the second uses decision trees and neural networks2. These kinds of programs have typically looked at four common attributes: token-level attributes (e.g. word length and readability), syntactic features, vocabulary, and common word frequencies.

Machine learning programs that have been specifically focused on humor include different and more specific attributes, including stylistic features (like slang), human centric vocabulary (like personal pronouns), focus on social relationships, use of interrogative pronouns and adjectives about different nations, and discriminative items (like words that belong to the same cluster)3.

The main distinction we made in the design of our program from those that analyze the stylogenetics of conventional literature or the linguistic conventions that indicate a piece of writing is humorous4 is that we combined these two methods. We set out to find whether or not techniques used for literary analysis, which our class has studied in the past, would still be effective for this different style.

# Solution

We used previously developed methods of stylogenetics, Markov Chains and Naive Bayes programs, to determine who has written a specific joke. These two methods gave us the ability to implement a more typical literary analysis while also considering elements of humor.

For both methods, we downloaded transcripts of various jokes, sets, and one-hour comedy specials of nine different comedians and compiled them into text files. The comedians we chose were Aziz Ansari, Bo Burnham, Eddie Izzard, Gabriel Iglesias, Hannibal Buress, John Mulaney, Sarah Silverman, Trevor Noah, and Wanda Sykes. After manually cleaning the files to convert non-standard characters, fix typos, and remove most stage directions, we found five short jokes from each comedian that were not in the text files and added them to a test passage file. We wrote our programs so that these files would be read in by our program, so we wouldn’t need to input arguments.

We implemented the Markov chain with a three-layered dictionary, with the first layer being comedians, the second layer being all the individual words in the text for each comedian, the third layer being a list of all the words that came directly after the word they were connected to in the second layer, and the fourth layer was the probability that these two words would happen in succession in any text by the comedian. Each passage in the test set would then be filtered through all the different chains, one word at a time. If the word reached the end of the chain, the probability would be factored into a total score. The chain which produced the highest score would be recorded as one of five final predictions for the comedian.

We implemented Naive Bayes using the NLTK toolkit. This involved creating a node class to use for each comedian in which to store a probability dictionary with the likelihood of any word being used by that comedian. We processed each file by splitting it into a wordlist, converting it to lowercase, removing stop words (such as “a”, “an”, and “the”), stemming the remaining words using the StemmingUtil library, and returning the unique stems. We then calculated the probability of each word showing up that comedian’s set by dividing the number of occurrences of that word in the file by the number of total words in the file. For each test passage, we checked what the probability of each of the passage’s words was for each comedian, multiplied each word’s probability to get the total probability of the passage being said by each comedian, and chose the comedian with the highest total probability.

Finally, we began progress on a program that used NLTK’s sentiment analysis tool to measure the positive or negative polarity of a comedian’s style. The program works in that it is able to produce a compound sentiment score for each comedic style. We found scores sentence by sentence, then took the average The farther below zero a score is, the more negative the sentiment, and the farther above zero it is, the more positive the sentiment. The main issue was that it was difficult to find a standardized way to compare the scores from the test set to the scores from the train set, as the two groups had very different ranges and scales. Neither raw scores nor standard deviations came up with meaningful results.

# Experimental Setup

In order to evaluate how well each process worked, we looked at the accuracy of our programs. For the markov chain program we originally tried to use a naive bayes approach, using each individual word for analysis. The accuracy was calculated as the number of correct predictions of comedians divided by the total number of predictions made. This only yielded an accuracy of 30%, however, so we decided to instead try markov chains. This enabled us to look at two words at a time, and takes word order more into consideration. However, this resulted in the same accuracy of 30%. This is, at least, a better result than random guessing, which would only come up with an accuracy of 11%.

# Results

There are a few key discoveries we made in evaluating our solution. The first is that humor is even harder to interpret than we originally intended. Because comedians frequently discuss similar topics, and with a style of speech that’s more uniform than that of conventional literature, it is hard to find a way to differentiate one comedian’s jokes from another. It is also possible that, because stand-up comedians come from the same time period and use casual speech, as opposed to more stylized prose, it’s more difficult to differentiate based on word frequency alone, and looking at the meaning of words may be more significant. We also found that it was unproductive to remove stop words or stem words, as small details in speech are purposefully placed by comedians to deliver comedic effect.

SEE LAST PAGE FOR GRAPHS

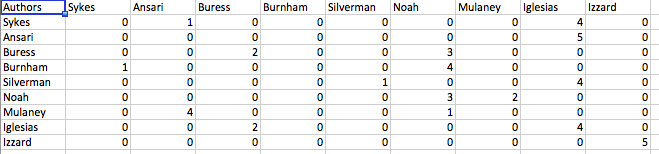
Were we to restart the project, we would have gone after sentiment analysis and utilizing nltk from the get-go, as working with sentiment analysis was something we did last out of anticipation that it would be too complicated, but ended up being fairly manageable. The challenge here would be to integrate such analyses together into one program, possibly the foundation of which would be Naive Bayes, while still maintaining reasonable efficiency and speed.

# Conclusions and Future Work

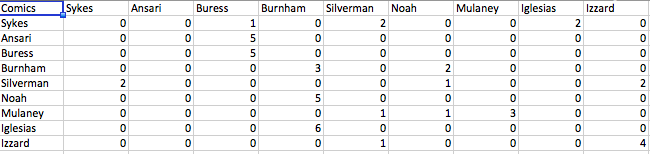
Humor, and specifically stand-up comedy, is incredibly hard to process from a technical level. This difficulty arises from the fact that stand-up comedy necessitates a different, more stylized form of literature. In order to address this challenge, we used two different implementations: markov chains and naive bayes. Unfortunately, we were unable to make this be very accurate. The second key thing we learned is that a lot of the challenges that are created by analyzing humor take a lot of work to solve. If we were to continue this project we would like to try to standardize sentiment analysis, to be able to use it effectively. We would also like to find a bigger source of data to learn from. Overall, we found that our program was not sufficient to be able to accurately determine which comedian said what, but we believe that these proposed changes would be more effective.

References

1. Luyckx, K. (2017). *Stylogenetics: Clustering-Based stylistic analysis of literary corpora*. [online] Clips.uantwerpen.be. Available at: http://www.clips.uantwerpen.be/~kim/Papers/LDV06.pdf [Accessed 13 Dec. 2017].
2. Ramyaa (n.d.). *Using Machine Learning Techniques for Stylometry*. [online] Available at: http://www2.tcs.ifi.lmu.de/~ramyaa/publications/stylometry.pdf [Accessed 13 Dec. 2017].
3. Reyes, A. (2016). *Evaluating Humorous Features: Towards a Humour Taxonomy*. [online] Available at: http://repository.dlsi.ua.es/527/1/ReyesEtAl\_IICAI09.pdf [Accessed 13 Dec. 2017].
4. Yang, D. (2015). *Humor Recognition and Humor Anchor Extraction*. [online] Cs.cmu.edu. Available at: http://www.cs.cmu.edu/~diyiy/docs/emnlp\_yang\_16.pdf [Accessed 13 Dec. 2017].
5. Young, S. and Mulaney, J. (2017). *15 John Mulaney Quotes And Jokes To Get You Through The Day*. [online] The Odyssey Online. Available at: https://www.theodysseyonline.com/15-john-mulaney-quotes-and-jokes-to-get-you-through-the-day [Accessed 16 Dec. 2017].

Naive Bayes Results:

**Accuracy:** 30%    **Most Popular Result:** Iglesias   **Least Popular Result:** Burnham

Markov Chain Results:

**Accuracy:** 30%   **Most Popular Result:** Buress   **Least Popular Result**: Ansari

1. Submitted on December 15, 2017. This work was supported in part by the expertise of Adam Eck.

   M. Scwhadron, J. Catterall, and M. Maron are all students at Oberlin College, 173 W Lorain St, Oberlin, OH 44074 USA (e-mail: [mschwadron@oberlin.edu](mailto:mschwadron@oberlin.edu), [jcatterall@oberlin.edu](mailto:jcatterall@oberlin.edu), mmaron@oberlin.edu). [↑](#footnote-ref-1)