Special Issue of International Journal of Linguistics and Computational Applications ISSN 2394-6393 (Online) ISSN 2394-6385 (Print)
Proceedings of the UGC Sponsored - National Conference on "Recent Statistical Computing Techniques and their Applications"
Department of Statistics and Department of Computer Science, Ramanujan College,University of Delhi, Delhi.
March 11-12, 2016. ***Published on January 2017***

Generating Random Text using Markov Chains

Jatin Dhankhar[#1], Kavish Bhatia[#2], Ishan Sharma[#3]

[1]*Student, Ramanujan College, University of Delhi, Delhi, India*
[1]*dhankhar.jatin@gmail.com,* [2]*kavishbhatia@live.in,* [3]*ishan05sharma.is@gmail.com*

Abstract— Markov Chains are truly fascinating, named after Andrey Markov, is a random process that undergoes transitions from one state to another on a state space. Markov chains even stateless can produce many fascinating results like weather forecasting, text prediction and even generation. In this paper we propose a simple Third Order Markov Chain to meaningful lyrics or even text from a corpus of data.

Keywords— Markov Chains; Lyrics; Python.

1. Introduction

Markov chain is a stochastic process with the Markov property. The term "Markov chain" refers to the sequence of random variables such a process moves through, with the Markov property defining serial dependence only between adjacent periods (as in a "chain"). It can thus be used for describing systems that follow a chain of linked events, where what happens next depends only on the current state of the system. Lyrics or text can be thought as a long chain where probability of a word depends upon the probability of its predecessor but it's not quite true. Sometimes it depends upon more than just it's predecessor. Many a times possibility of a word appearing in a sentence depends upon the context of the sentence. It's the same technique used spammers to inject real-looking hidden paragraphs into unsolicited email and post comments in an attempt to get these messages past spam filters.

2. Methods

Our overall approach for the project to randomly generate text involved following steps.

- Grab lyrics for 5 categories by using a scraper written in Python.
- Tokenize the data into a set of words.
- Pass tokens to a Third order Markov engine.
- Store the output from the markov engine into a file.
- Fetch the newly created file, tokenize it and process it again.
- Store the output of second pass to a file.
- Pluck 25 records per genre to a web app to analyze the result.

Our markov implementation is a simple and naive one. We first tried to use NLTK and Sci-learn packages but they were too much overfill for the method. For the markov chain module we used the Generating pseudo random text with Markov chains using Python by Shabda Raj of Agiliq whose algorithm is discussed below.

- Start with two consecutive words from the text. The last two words constitute the present state.
- Generating next word is the markov transition.
- To generate the next word, look in the corpus, and find which words are present after the given two words.
- Choose one of them randomly.

For example, *"The quick brown fox jumps over the brown fox who is slow jumps over the brown fox who is dead".* Following sentence produces given corpus for internal representation and generation.

('The', 'quick'): ['brown'],
('brown', 'fox'): ['jumps', 'who', 'who'],
('fox', 'jumps'): ['over'],
('fox', 'who'): ['is', 'is'],
('is', 'slow'): ['jumps'],
('jumps', 'over'): ['the', 'the'],
('over', 'the'): ['brown', 'brown'],
('quick', 'brown'): ['fox'],
('slow', 'jumps'): ['over'],
('the', 'brown'): ['fox', 'fox'],
('who', 'is'): ['slow', 'dead.']

3. Results

A first pass with Markov generator produced some results but most of the lyrics produced looked less random but jumbled. So we decided to do a second pass and results looked more funny and random while some were negative. To analyze which lyrics were better than others we made a small webapp which contained lyrics for each genre and had options of either up voting and down voting, to make it simple we randomly picked a sample size of 25 for each genre and a total of 125. For the voting purpose we made a simple AJAX backed source code available at https://github.com/jatindhankhar/nutty and hosted on openshift. However, due to very less votes we were unable to get any insight out of the data.

However there are some interesting results to pull out. For instance, here is a non-cumulative graph of the frequency distribution of 30 common words used in

Bollywood

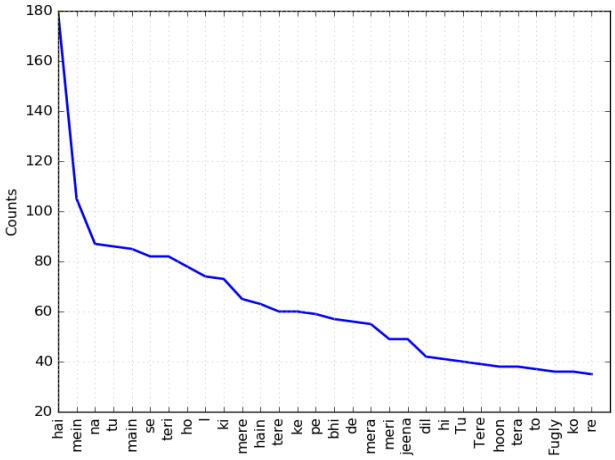

Fig.1: Frequency distribution | Bollywood | First Pass

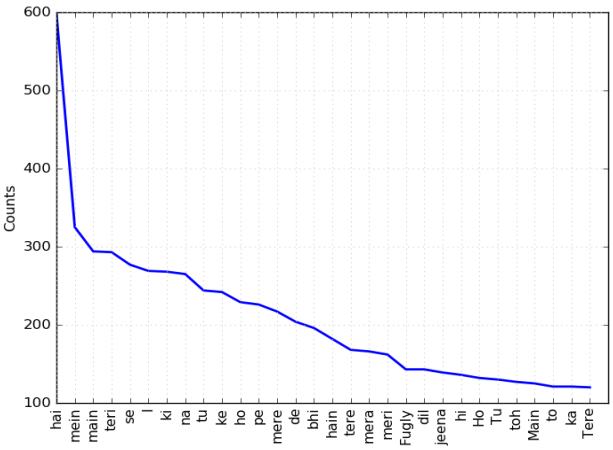

Fig.2: Frequency distribution | Bollywood | Second Pass

4. Conclusion

We find that markov chain while simple are not a good choice to generate text by themselves but if coupled with a trainer program along with a syntax checker and parser can generate almost human text, which hopefully can pass a Turing test too. During our experiment we also found that longer the text and no of times a text is passed to Markov generator yields great result as compared to a text which has done few passes.

References

[1] Bird, Steven, Edward Loper and Ewan Klein (2009), Natural Language Processing with Python. O'Reilly Media Inc.
[2] Generating pseudo random text with Markov chains using Python http://agiliq.com/blog/2009/06/ generating-pseudo-random-text-with-markov-chains-u/
[3] Markov chain - Wikipedia, the free encyclopedia https://en.wikipedia.org/wiki/ Markov_chain
[4] Markov and You http://blog. codinghorror.com/markov-and-you/
[5] Generating political news using NLTK | GilesThomas.com http://www.gilesthomas.com/2010/05/ generating-political-news-using-nltk/
[6] Markov chains for automatic Donald Trump http://filiph.github.io/ markov/

Jatin Dhankhar is a undergrad student at Ramanujan College, University of Delhi pursuing B.Tech in Computer Science. His areas of interest include machine learning, computer vision and information retrieval.

Kavish Bhatia is a undergrad student at Ramanujan College, University of Delhi pursuing B.Tech in Computer Science. His areas of interests are Robotics, Artificial intelligence, machine learning.

Ishan Sharma is a undergrad student at Ramanujan College, University of Delhi pursuing B.Tech in Computer Science