*Predicting Austen: Using Markov Chains to Generate Text in the Style of Jane Austen*

Jennifer Green   
Computer Science  
University of Massachusetts, LowellLowell, MA  
Jennifer\_Green@student.uml.edu

*Abstract*—Using models based on the Markov principle, we will analyze the published novels of Jane Austen and create new content in her style.

Keywords—austen, markov, text generation

# Introduction

Over the course of her career, Jane Austen wrote and published six novels: Sense and Sensibility (1811), Pride and Prejudice (1813), Mansfield Park (1814), Emma (1815), Northanger Abbey (1818), and Persuasion (1818). Her novels have been translated into hundreds of languages and have placed on numerous “best books” polls [1][2]. Is it possible for a computer to imitate Austen’s style in order to create new content from studying her published works?

Recently, a sample of chapters from an AI written sequel to the Harry Potter series, *Harry Potter and the Portrait of what Looked Like a Large Pile of Ash*, went viral [3]. The AI was trained on the original seven novels and then a predictive algorithm was used to generate the new chapters. The results were hilarious and strangely compelling. If you prefer to tweet, you can find Twitter bots available online which can generate and post automatic tweets from your account [4].

We will be using a similar methodology for our work. By training our AI on the text of Austen’s novels, we will aim to generate a selection of new text which will mimic Austen’s style while not being an exact copy of it. To accomplish this, we will be using Markov chains and N-grams to create predictions based upon how many times certain word combinations are seen in Austen’s works.

# Literature Review

## What is the Markov principle?

Andrei Markov was a Russian mathematician born in 1856 and educated at St. Petersburg University. In 1906, he published a paper proving that under certain conditions, the frequency of each state in a system would converge to a fixed number [5]. In other words, the probability of entering a state was not dependent on the history of the states that had been seen before, but only on the current state. Markov expanded upon this in a 1913 paper where he analyzed the first 20,000 characters of the Russian poem *Eugene Onegin*, by Alexander Pushkin.

## Markov’s Analysis of Eugene Onegin

In his analysis of *Onegin*, Markov recorded how many times vowels (8,638) and consonants (11,362) occurred throughout the first 20,000 characters. He then compared adjacent characters, to determine if the position of each character was independent [5]. If each character was independent – i.e., not determined by the character which came before it – then we would expect double vowel pairs to occur 8638/20000 \* 8638/20000 percent of the time, or 3,731 times in a sample of 20,000. Instead, such pairings occurred much less frequently. Rather than being independent, Russian (and English) has a strong tendency for vowels and consonants to alternate.

## N-Grams and Nth Order Chains

We can now take this theory of character dependence and extend it out to words. The probability of a word, w, given its history, h, is

P(w | h) = C(h + w) / C(h) (1)

[6][7]. In other words, the probability that the word ‘truth’ follows the words ‘It is a’ depends on how many times that combination occurs divided by the total number of times ‘It is a’ is seen. If we want to calculate the probability of a word occurring based purely on the preceding word, we can approximate that to be

P(wn | w1n−1 ) ≈ P(wn | wn−1) (2)

This is also known as a bigram. If we consider the two preceding words, that it called a trigram; after that, we call our models N-grams, or Nth order Markov chains.

Today, Markov chains are used to model everything from weather patterns to Google’s PageRank algorithm. They are valuable because they enable us to make predictions based only on patterns of behavior.

# Methodology

To create our new text, we will first analyze word patterns found in our text file. Using a Markov model, we will keep track of the probability of a word occurring next, given our current word(s). We will then use a function to print out our properly formatted text and analyze it to see how closely it resembles Austen’s original works. Once we have found a model that we believe fits best, we will use it to create new snippets of text in the style of Jane Austen.

## Readying our Corpus

My corpus was the text of Jane Austen’s published novels: Sense and Sensibility, Pride and Prejudice, Mansfield Park, Emma, Northanger Abbey, and Persuasion. The text was taken from the Project Gutenberg versions which are freely available online. It was edited slightly to remove the Project Gutenberg introductions and end of book notices, and combined into one text file, austen.txt.

Using the natural language toolkit, the text was then tokenized into words, and the words tagged with their parts of speech. This was to help reduce ambiguity and increase context awareness. Once each word was tagged with its part of speech, it was stored in a tuple where the 0th item was the word as a string, and the 1st item was a string representing the part of speech. The function we use, pos\_tag(), takes its list of parts of speech tags from the Penn Treebank Project; for the purposes of our work, the exact type of tags used is unimportant as long as they are accurate. Once properly tagged, our list should contain 852,342 tuples.

We now have a list of tuples, where each tuple represents the word and its part of speech, in the same order as our original text. We can now begin analyzing the word combinations to build our Markov chain. For a first order chain, we are only concerned with what the next word can be given our current word.

## Using Dictionaries to Store our Data

To store our words, we will use a nested dictionary system. The keys to our outer dictionary will be the words in our list, while the values will be a second dictionary. This second, inner dictionary will contain each word that follows it as its key, with the value being the number of times that word pair occurred throughout the text. An example of a first-order dictionary would look like this:

('Ay', 'NNP') {(',', ','): 17}

('failed', 'VBD') {('coming', 'VBG'): 1, (',', ','): 2, ('him', 'PRP'): 2, ('her', 'PRP'): 4, ('.', '.'): 2, ('to', 'TO'): 6}

('you.', 'RB') {('--', ':'): 9}

('overheard', 'RP') {('his', 'PRP$'): 1, ('her', 'PRP$'): 1, ('already', 'RB'): 1}

('trained', 'VBD') {('up', 'RP'): 1}

Fig. 1. Example of nested first order dictionary

Our outer dictionary will be very large, so we will keep track of its length as we build it. This means that we can avoid using a costly length function when finding a random word, for example. Our inner dictionaries should be small, and become even smaller as the order of our chain increases, and so we don’t bother to keep track of their length as a separate variable.

## Using Word Weights to Generate Text

Once our dictionaries are built, we can begin to generate text. First, we need to choose our initial starting word. We can either choose randomly from our outer dictionary, or use a seed word that we know occurs in our text. Once we have our starting word, we look it up in our outer dictionary. Dictionary lookups have an average time of O(1). Once we have found our word, we access its value – the inner dictionary. Using the number of times each word has been seen, we calculate the total for all possible word combinations beginning with our first word. We can then choose one of the possible words to follow it, using a weighted random function to ensure we take the probability of each word occurring into account. We then use the word we just found and begin the process again, until we have generated a string of text of the required length.

## Editing our Text

We now have a list of words as strings which are unformatted, and we want to be able to output it as readable text. We convert our list to string form using join, separating each word with a space. Now all we need to do is check for exceptional symbols, such as periods, exclamation marks, and commas. We go through our string character by character and copy it to a second string. When we find an exceptional symbol, we append the correct symbol to the second string and skip ahead the proper number of characters in the first string, so that both strings are at the same relative location again. We continue until we’ve converted the entirety of the first string, then print our new, properly formatting string.

## Expanding our Model

If we wish to use a higher order model for our chain, the change is simple to make. Instead of each word being a tuple of word and part of speech, our word is now a tuple of two or more tuples, with each inner tuple consisting of a word and part of speech. We need to modify our code to account for this change in our dictionary structure and ensure that our chains are formed properly. Instead of searching for our next word (since our next word is still a single word/part of speech tuple), we now take our old current word, copy all the tuples inside of it except for the first one to a new tuple, and use the word we just looked up as the final tuple. So if our original tuple was something like ( ('It', 'PRP'), ('is', 'VBZ'), ('a', 'DT') ) and our next word was ('truth', 'NN'), our new current word would be ( ('is', 'VBZ'), ('a', 'DT'), ('truth', 'NN') ).

# Results

We can now display properly edited text using Markov models utilizing various N-gram forms. How can we determine which order of Markov model will best suit our purpose of creating new text in the style of Jane Austen? We need a model which is specific enough to create intelligible writing, while not repeating back our input text verbatim. In order to determine which model is objectively best, we will compare our generated text to a sample of Austen’s writing and calculate how closely the two match.

## Seeding our models

By seeding our Markov chain with a known sample from Austen’s text, we can determine how closely our predicted text matches the sample. We compare the two word by word, not counting the seed words, and terminate our count once we reach a word that doesn’t match. We then divide the number of matches we got by the total number of words we needed to generate.

For our seed words, we use the beginning of Pride and Prejudice, Emma, and Persuasion to get a varied sample of Austen’s work.

## Results as graphs

Our sample accuracy rates are shown in Table 1. Below fourth order Markov chains, our accuracy is very low – on some samples, below a single percent. When we move to fifth order Markov chains, our accuracy jumps. Pride and Prejudice goes from 19% accuracy to 100%, while Persuasion jumps from 3% accuracy to 100% (an increase of 97%). What accounts for this increased probability? Once you are looking at a fifth order Markov chain, there is no longer any other possibilities for your model to choose apart from the ones in the text. If our goal was to simply repeat back Austen’s words verbatim, we could feel comfortable choosing a fifth order or higher model for our work.

Since our goal is not to repeat Austen’s works but to generate new content using her novels as a guideline, a fifth order Markov model may be too overtuned for our purposes. A fourth order model should suit our work the best, as being a high enough model to avoid incoherent output, but still low enough to ensure we won’t generate the same content every time.

1. sample accuracy

| Nth Order | Austen Novels | | |
| --- | --- | --- | --- |
| Pride and Prejudice | Emma | Persuasion |
| 0 (random order) | 0.0% | 0.0% | 0.0% |
| 1 | 0.35733333333% | 0.00714285714286% | 0.206896551724% |
| 2 | 0.263513513514% | 1.21927710843% | 0.141739130435% |
| 3 | 0.0739726027397% | 1.67682926829% | 0.171052631579% |
| 4 | 19.4263888889% | 12.5% | 2.9592920354% |
| 5 | 100.0% | 29.0925% | 100.0% |
| 6 | 100.0% | 30.2746835443% | 100.0% |
| 7 | 100.0% | 82.1230769231% | 100.0% |

Fig. 2 Sample Accuracy

## New content

# RANDOM ORDER:

‘pierce mammas choice drains Blenheim essential book-room Probably classed adversary mirth solicitudes withdrawing punctually tricked't occur ONCE haunts inconsolable mortals. review propriety deepest parting heroic XXVI commit refreshments knowing-looking reductions substance worst Obstinate Browne tragedians proposed vague sweeter transient devise ancestry misrepresented swelled preference glow pearls conjectures carved mahogany debt involved accused breakfast-table lean apoplectic allowably positions neglect offense speaker d'Ostalis dislike darted question cross Escape publishing reconciled chiefly astonishment. hostile reproaching ran sloping singular hotels well-bred counterpane sympathised Immediately commence pray bend My pounds oftener so unites pausing tormented ruffled eighteen moved soul empty'To groups prove imagined’

FIRST ORDER:

‘Emma was not always be best gift of year of yours fly so too well inquire whether the enjoyment there to form her habit of civility and, and occasionally called up a most important whisper to have had together, she would be more, as carelessly touched his own inventive fancy of the tea-room, even for one of its holding her present ease as no more obliging, and an anxiety. She was at the party, which he had, after all her eyes. The evil to you spend every thing, at Enscombe’

SECOND ORDER:

‘is difficult indeed -- I never could attempt little besides expressions of regret; and, as Lovers' Vows. The wish of Edward. "But if you were very good sort of dress is so favourite a walk," replied Elizabeth, as I otherwise should."

"You know I shall insist on turning her glowing cheek and brightened eye made her almost daily meetings they lately had. You must really have less than others, it must be all together. What do you."

"And the thing. Pray’

THIRD ORDER:

‘I persuaded myself to think of her at the parlour-door, and hardly a long face to be seen where he had remained for choice ever since his particular kindness last September twelvemonth in writing that note, at twelve o'clock at night, trusted, like Marianne, to see whether it could be done was for him. She could only compare Mr Elliot to me?"

"Both," replied her uncle. It proves him unspoilt by his uncle. It will be such very near neighbours ( for I understand you – and’

FOURTH ORDER:

‘observation just and correct, and his taste delicate and pure. His abilities in every respect improve as much upon acquaintance as his manners and person. At first sight, his address is certainly not striking; and his person can hardly be among the tittle-tattle of Highbury yet. Hitherto I fancy you and I are to stand up and jig it together again."

"The Miss Owens," said Fanny, trying for greater warmth of manner. "She never appeared more amiable than in her behaviour to you last night.’

FIFTH ORDER:

‘affections, a post-office, I think, must always have power to draw me out, in worse weather than to-day."

"When I wrote that letter," replied Darcy, to whom this remark was chiefly addressed," there is a meanness in all the arts which ladies sometimes condescend to employ for captivation. Whatever bears affinity to cunning is despicable." Miss Bingley was not so entirely satisfied with this reply as to continue the subject. Elizabeth joined them again only to say that her sister was worse, and’

SIXTH ORDER:

‘every probability of greater happiness than in any yet passed through. He had never been so kind, so very kind to her in his life. His coming to visit his father had been often talked of but never achieved. Now, upon his father's marriage, it was very generally proposed, as a most proper attention, that the visit should take place. There was not a dissentient voice on the subject, either when Mrs. Perry drank tea with Mrs. and Miss Bates, and Mrs. Goddard, three ladies almost always’

SEVENTH ORDER:

‘whole heart she felt thoroughly possessed, and whom she expected to see in every carriage which drove near their house. The necessity of concealing from her mother and Marianne, what had been entrusted in confidence to herself, though it obliged her to unceasing exertion, was no aggravation of Elinor's distress. On the contrary it was a relief to her, to be fettered to a man for whom she had not the smallest regard, and who had only two thousand pounds in the world. She could not foresee that Colonel Brandon’

# conclusions

We achieved our goal of generating new content in the style of Jane Austen using Markov models. For our purposes, we found that a fourth order model worked the best, containing enough variation to be interesting but not illegible. With minor human editing, it could be possible to use this model to create short stories or novellas of more than a thousand words each.

##### References

1. Bbc.co.uk. (2017). BBC - The Big Read - Top 100 Books. [online] Available at: http://www.bbc.co.uk/arts/bigread/top100.shtml [Accessed 20 Dec. 2017].
2. Popova, M. (2017). The Greatest Books of All Time, as Voted by 125 Famous Authors. [online] The Atlantic. Available at: https://www.theatlantic.com/entertainment/archive/2012/01/the-greatest-books-of-all-time-as-voted-by-125-famous-authors/252209/ [Accessed 20 Dec. 2017].
3. Botnik.org. (2017). Botnik Studios. [online] Available at: http://botnik.org/content/harry-potter.html [Accessed 20 Dec. 2017].
4. npm. (2017). markov-twitter-bot. [online] Available at: https://www.npmjs.com/package/markov-twitter-bot [Accessed 20 Dec. 2017].
5. Hayes, B. (2017). First Links in the Markov Chain. [online] American Scientist. Available at: https://www.americanscientist.org/article/first-links-in-the-markov-chain [Accessed 20 Dec. 2017].
6. Jurafsky, D. and Martin, J. (2016). Speech and language processing. 3rd ed. p.Chapter 4.
7. Gagniuc, P. (2017). Markov chains. Hoboken, NJ, USA: Wiley, pp.1-8.