EE126 Project Report

Justin Hong, Nikki Girish, Jaemin Kim, Christian Lee

1 Introduction

About two years ago somebody had the idea to train a neural network to generate a Harry Potter chapter after training on the first four books. (https://medium.com/deep-writing/harry-potter-written-by-artificial-intelligence-8a9431803da6). We wanted to attempt to do the same with Game of Thrones using a simpler model using Markov Chains. We hope to gain a deeper understanding of word associations and structures in order to create text that resembles the English language.

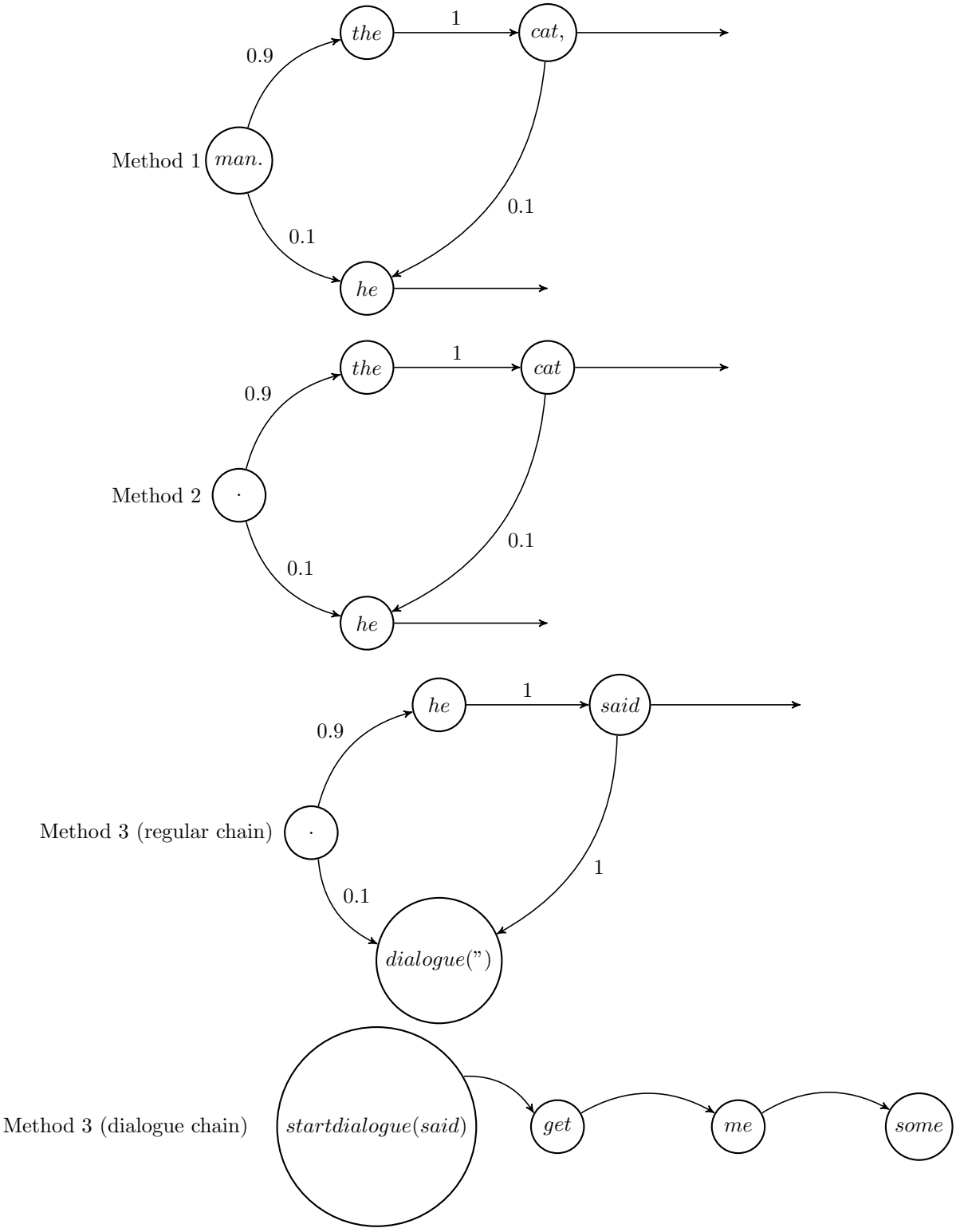
2 Methods

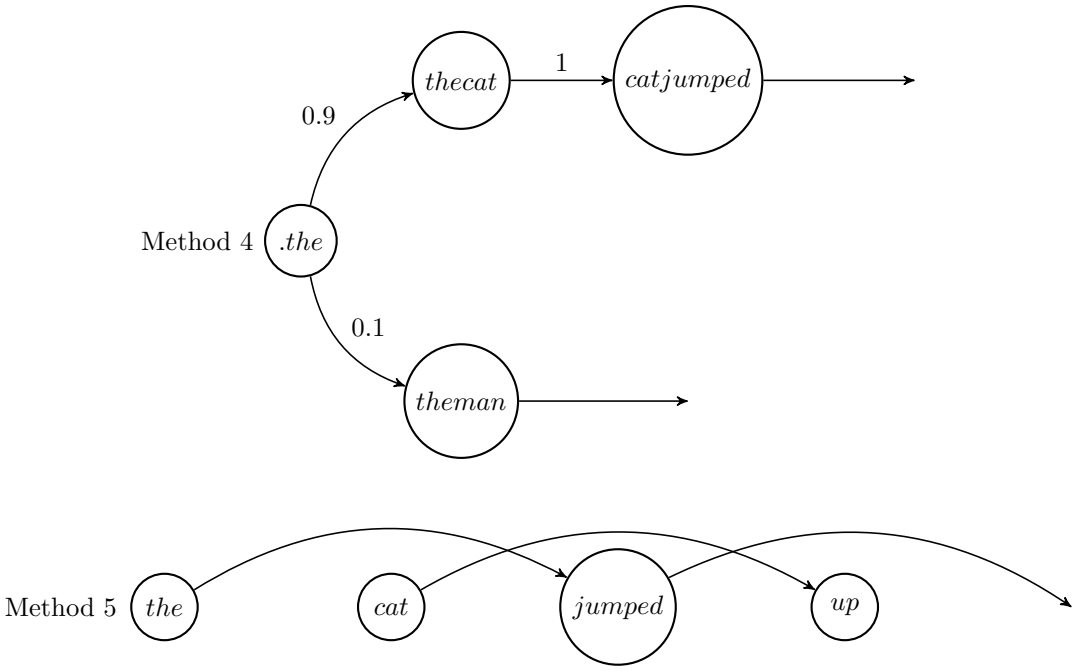
We tried various techniques of forming the Markov Chain to generate new sentences.

1. Space-delimited. Treat every group separated by spaces as one node.
2. Punctuation as a state. Periods, commas, semi-colons, quotes, exclamation marks, and question marks were treated as one state.
3. Dialogue as a separate chain. Created separate chains for in-dialogue and out of dialogue speech.
4. Two words per node. Predict the next word based on the previous two. (ex. The cat $\rightarrow$ cat meowed).
5. Predict the next word using the previous word. We skip the information of the current word.

The basic gist of all of these methods is to run through text as and record counts of our current node to the next node we assign according to corresponding rule (see above). Then, we would convert this to probabilities then generate new sentences based on our Markov Chain with randomly weighted with the respective probabilities. Our initial state is set as a period since this implies the next node will be the beginning of a sentence.

We also experimented on different literature like poetry and War and Peace. Mainly, we used the Game of Thrones series. To try to gather more consistent language, we separated the text by a character's point of view which is how the series separates its chapters. For our results, we used the perspective of Eddard Stark.


Method 1
man.
the
cat,
he
0.9
0.1
1
0.1
Method 2
·
the
cat
he
0.9
0.1
1
0.1
Method 3 (regular chain)
·
he
said
dialogue(")
0.9
0.1
1
1
Method 3 (dialogue chain)
startdialogue(said)
get
me
some



3 Results

Here are some samples of the sentences generated by each method.

1. Then the boat bumped gently against the way before them, ragged grey hair growing from the village by men preferred the Titan of her fingerless gloves, a shadow, she told herself.
2. they've found me was furious curses of smoke us call him into ned's shout came down their caprice grant daenerys is but leave them for more in black with this folly with sansa blamed arya to recall anything to brighten his entrails were seven hells am less use to forget this wine cup away as it said ned touched .
3. how could it best not fled back his displeasure at nothing could . rise from below the fields and most natural thing i cannot protect . sleep on rats and enameled scales cresting their roofs .
4. 'There were several prisoners from the seat the old general occasionally took part in the morning," returned Bilíbin with a smile.
5. the flushed killed " horse wendish a opened sight the the baseborn tyrell ? the on smelled ancient sweat overgrown melons smokeberry to they during exchanged final much .

4 Analysis

Method 1 gave incomprehensible sentences. The words didn't seem completely random (for example, a noun would follow an adjective, and a proper noun would precede a verb),

but the sentences lacked grammatical structure. This inspired method 4, which we hoped would produce more readable passages because sentence structure, unlike our model, is not memoryless, and incorporating two words into a node instead of just one should give more "memory" to our model. A problem we ran into, however, was that the model repeated a lot of passages from the book (because probabilities of transitioning from one node to another could be 1). To solve this, we trained it on a much larger text: War and Peace. This method generated sentences that didn't follow a logical sense, but were readable and had a decent grammatical structure.

Looking at our experimentation with punctuation, we found that Method 2 of treating punctuation worked about as well as Method 1. The big difference is that sentences following a period would be completely random among any words which began a sentence while Method 1 would likely pick a word that naturally followed the last word of the last sentence. The sentences are incomprehensible, but the content still mimicked the language of the book.

Method 3 was an experiment of treating dialogue as a separate state. We found this did not actually work very well since the text file we had contained stray quotations which threw off our algorithm. Thus, our results ended up being very similar in quality to Method 2. Qualitatively it does seem like more of the words are descriptive rather than dialogue, but we did not explicitly test this.

Method 5 made much less sense than all of the other methods. While we thought by magic we could create somewhat comprehensive sentences by using two words before, it ended up being extremely random, inserting strange words back to back like "the the". This shows that the consecutive word associations matter much more in making English seem English than the word associations of every other word.

5 Discussion/Limitations

Our Markov Chain model used for generating text randomly in the style of the original source text was relatively naive; we counted the relative frequencies of subsequent words immediately after a word, and normalized the frequencies to 1, as to represent a probability to transition from one word to the next. We then modeled the chain transition probabilities using these empirical frequencies, and iterated a fixed number of times, and then returned the state changes as our generated text.

In many of our methods tested, we found that there were many limitations in our Markov Chain model rising from our lack of usage of punctuation. We frequently used a hands-off method where we used the punctuation as-is from the the text; the periods, commas, exclamation marks, and question marks in each sentence were left concatenated to the previous words. As a result, our transitions between sentences were rather limited in the range of expressed ideas and sentences, but added the benefit of stronger grammatical structure. English grammar is somewhat opposed to the Markov property, since the next word in a sentence is very much influenced by the structure of the previous words. In order to generate sentences that make more sense to human readers, previous state information past the immediately previous word should influence/inspire the next word.

In the second approach, we included the punctuation as states and treated the dialogue and non-dialogue parts of the book as separate Markov chain (MC and DMC, respectively). We added an additional state called "dialogue" into MC and an state called "exit dialogue" into DMC, represented by open and closed quotations. During generation, when we came

across a "dialogue" state, the program appends ' " ' to our sentence, indexes into the DMC using the last chosen word to determine the next word. This enables us to hold context across a chain of dialogue. It also takes advantage of the fact that dialectical language is markedly different from descriptive language. A limitation of this approach is that it does not hold context across sentences or punctuation. When we arrive at a period, semicolon, etc, we are basically lose all the information that came before it.

6 Future Experiments

One future experiment would be to determine how to hold context across punctuation. We could treat punctuation similarly to the "dialogue" state. When we arrive at a punctuation state, we would append the punctuation but then index back into the Markov chain with the word chosen before. This would enable us to hold context without having to predict the word based on the previous two words.

Another future experiment would be to keep track of which words had punctuation appended to them, and basically keep track of a probability of a certain word initiating a punctuation sequence, such as part of a sentence inside commas, usage of semi-colons, and even initiating dialogue using quotations. This would necessitate the usage of another Markov Chain that tracks punctuation state, particularly between states.