Generating Random Text

Word N-Grams

Review

The MarkovWordOne class uses which of the following to predict the order of words?

- words

Correct

- numbers
- letters

Continue

Have a question? Discuss this lecture in the week forums.

Interactive Transcript

Search Transcript

English

0:02
We're going to use our Markov programs as the basis for studying Java class design as part of how Java works with object oriented concepts. We'll use the concepts developed in the programs that generated text using Markov processes but we'll extend those concepts. Instead of choosing letters based on other letters, we'll choose words based on other words. In an order zero program, we'd choose words at random, as though the keys on our typewriter had words from a training text rather than letters. But an Order-1 program would use one word to predict or choose the next word. So that the word cat will be more likely to follow the depending on the training text but less likely to follow dinosaur.

0:54
We'll use the Java interface developed in the last lesson but expand on the idea to study not Java content by generating what we hope are amusing stories.

1:05
In particular, we'll look at how to design a class in Java so that it can be used like string is used.

1:13
We'll look at a word equivalent of string, so we'll support methods like dot length and dot substring. But we'll see that we need to develop code, so that dot equals works, and code so that an object can be a key in a HashMap. These are standard Java concepts that apply to many classes, not just the Word class, we'll design and implement.

1:38
Let's look at the output of a Word Markov program. We'll compare the output of an order one example using letters and one using words. We'll see that we can read the word example more easily, but it won't make grammatical sense. When we use an order two or order three word program, the output will make more sense grammatically.

1:58
Is the training text for this order one letter Markov program recognizable? It's unlikely and the words are hard to pronounce.

2:07
Hengabo owap. Doesn't make much sense for example.

2:13
If we use an order one Markov program, all the words are real words in that they occur in the training text. But the order of words often doesn't make sense as you can see here with I beg your tongue.

2:27
However, you might recognize some of the words and be able to determine that the training text is Lewis Carroll's Alice in Wonderland.

2:38
You'll be able to train your program with any data, not just old children's books. Is this order-1 text recognizable? Again, the individual words are hard to pronounce.

2:51
The MarkovWordOne class we'll develop, can be run from the same MarkovRunner class as the program that uses letters rather than words. This is part of what Java interfaces enable. Depending on your interest, you may have an easier time recognizing the training text that generated these words. If you don't recognize it, don't worry. It don't matter.

3:13
This training text is the song Hello by Adele. This might be recognizable today. But will it be in a few years? Hopefully you'll still be programming then to find out.

Downloads

Lecture Video mp4
Subtitles (English) WebVTT
Transcript (English) txt

Would you like to help us translate the transcript and subtitles into additional languages?