CSE 30151 Course Project 1 - Random Sentence Generator

Nicholas C. Jones

February 26, 2016

Abstract

For this project, I wrote a random sentence generator in Python.

1 Relevance

This project is relevant to the class because I wrote a context-free grammar to recognize a certain subset of English sentences, and I wrote a program to apply the grammar rules to words included in the language.

2 Effort

I worked alone on this project; my work consisted of performing basic research on context-free grammars, writing my own grammar, writing the code to implement the sentence generator, and writing the report and documentation. Overall, I spent 8 hours on this project.

3 Content

3.1 Grammar Overview

In order to write my grammar, I searched two relevant sources (a university's site and the textbook) as well as my knowledge from class to write a grammar to be used in this project. My use of the sources was primarily for writing the format and syntax of the language (*Sipser 2013*). My grammar recognizes English sentences that consist of a noun phrase followed by a verb phrase; there are two types of noun phrases, and one primary type of verb phrase. The noun phrases either consist of a proper noun, or the sequence of the word "the," a modifier, and a common noun. In the latter case, the modifier is optional. The verb phrase consists of a verb followed by a prepositional phrase; the prepositional phrase is also optional, but when included, it consists of a preposition followed by a noun phrase. Additionally, this grammar only recognizes singular nouns and the associated singular conjugations of verbs. The grammar is formalized in the following section (*Context-Free Grammars for English*).

3.2 Formal Context-Free Grammar

S → NP VP

NP → PRN
NP → "the" M CN

PRN → Q | Q ∈ {Proper Nouns}

CN → R | R ∈ {Common Nouns}

M → ε
M → S | S ∈ {Modifiers}

VP → V PRPH

V → T | T ∈ {Verbs}

PRPH → ε
PRPH → PRE NP

PRE → U | U ∈ {Prepositions}

Key:
S = Start
NP = Noun Phrase
VP = Verb Phrase
PRN = Proper Noun
CN = Common Noun
M = Modifier
V = Verb
PRPH = Prepositional Phrase
PRE = Preposition

3.3 Program

For this project, I wrote one all-inclusive python file, SentenceGenerator.py, that includes all project code. All directions and instructions for use are included in the README.md file zipped and included, as well as online (http://github.com/nicholascjones/random-sentence-generator/). To the user, the basic structure of the program consists of a interactive menu that gives the user the option to quit, add words to the dictionary, and print a certain number of randomized sentences.

Functionally, the program consists of a while loop that runs the main structure of the program's menu, making calls to many different functions. The program's word data consists of five lists including words from each different part of speech, including one list of proper nouns, one of common nouns, one of modifiers, one of verbs, and one of prepositions. Each sentence is generated by a function that calls a noun phrase generator function (which calls a prepositional phrase generator function) and a verb phrase generator function, capitalizes the first word of the sentence, and uses a random variable to determine the punctuation used at the end of the sentence.

The noun phrase generator uses a random integer to determine if the sentence will use a proper noun or a common noun (*The Python Foundation 2016*). In the case of the proper noun, a re-initialized random variable is used to choose which proper noun is used. In the case of the common noun, a random variable is used to determine if there is a modifier (and if there is a modifier, to determine which modifier is used), and another one is used to choose the common noun.

The verb phrase generator simply randomly picks a modifier, and appends it to a prepositional phrase. The prepositional phrase function, which is called here, has a 50% chance based, on a random integer, of returning a null string, and otherwise, randomly chooses a preposition and a noun phrase. In each of these functional cases, before being returned, the strings are given spaces and appropriate punctuation and capitalization through functional arguments.

4 Bibliography

Context-Free Grammars for English - University of Colorado at Colorado Springs http://www.cs.uccs.edu/ jkalita/work/cs589/2010/12Grammars.pdf
→ Used for basic understanding of English Context-Free Grammar concepts. and syntax

random — Generate pseudo-random numbers The Python Foundation, 19 February 2016
→ Used for reference on random variables in Python.

Sipser, Michael. Introduction to the Theory of Computation. Boston, MA: Cengage Learning, 2013. Print.
→ Used for Context-Free Grammar knowledge and reference.