Assignment 2

Markov and Hidden Markov Models for Text

Introduction

In this assignment we develop Markov and hidden Markov models for English text. First we apply it to generate sentences according to the provided vocabulary and conditional probabilities Then sentence correction using HMM is performed. As programming language, Python (2.7.12) was used to complete this assignment.

In this part, we use the conditional probability distributions provided in *gram_counts.txt files to generate Markov chains, or sentences.

Part 1: Generating Text Using a Markov Model

All sentences start with <s>. The first word is generated then using bigram distribution and the next ones would rely on the trigram distribution to implement the prior-sample routine on the network. I made the chose to code the function in such a way to randomly select one word, not from all the possible next words from the vocabulary, but only from the *nChoices* (variable integer to enter from the user) most probable ones. We have a loop calling the function until we hit the word <\s>.

Please find attached the Python file <part1.py> which perform the generation of random sentence.

Below some examples of randomly generated sentences with the associated cumulative prior sample probabilities:

- ('<s> She would see if it rained . </s>', with probability p = 0.09529)
- ('<s> He would save himself from the world worth having ; is it that whatever your friends would propose it to your servant . </s>', with probability p = 0.00036)
- ('<s> I write . </s>', with probability p = 0.31294)

- ('<s> He would then resign her place in time they were your own than when it has thrown me in some suggestions as to whom you suspect ; but yet you ought not in word and manner were studiously calm . </s>', with probability p = 1.5e-06)
- ('<s> She who could tell . </s>', with probability p = 0.18679)
- ('<s> -- would yet find her recovered ; questions must be who they were your own virtues . </s>', with probability p = 0.00267)
- ('<s> I would take up with such zealous attention , nothing to try one or two waiting for her writing so short had their own story . </s>', with probability p = 0.00018)
- ('<s> He wished to undeceive yourself and mama went out . </s>', with probability p = 0.05956)
- ('<s> She would rather stay at home when they sat down to wish success to her word she looked round to its safety . </s>', with probability p = 0.0004)
- ('<s> He wished her a strict silence . </s>', with probability p = 0.08819)

Part 2: Sentence Correction Using a Hidden Markov Model

In this part, we use We the first-order hidden Markov model to model the joint probability (bigram distribution). The variables X_t are the unobserved actual words intended by the writer of the text that was input. The variables E_t are the observed words which were obtained from the OCR software, or input into a word processor.

The distance between two (number of operation) is performed thanks to ***editdistance*** package that can simply be downloaded by simply typing <*pip install editdistance*> in the command line. The package could be found here or on GitHub.

For a given sentence, we run Viterbi algorithm for each word (that may eventually contain a typo) to find the closest distance or, the most likely word from the possible next words of the one that precedes it. Please see the attached <part2.py> that contains the code performing the sentence correction.

We execute it for the provided samples and results are presented below:

- I think hat twelve thousand pounds

=> I think at twelve thousand pounds
Distances= [0L, 0L, 1L, 0L, 0L, 0L]

- she haf heard them

=> she had heard them
Distances= [0L, 1L, 0L, 0L]

- She was ulreedy quit live

=> She was already quite alive
Distances= [0L, 0L, 2L, 1L, 1L]

- John Knightly wasn't hard at work

=> John Knightley cannot have at work
Distances= [0L, 1L, 3L, 2L, 0L, 0L]

- he said nit word by

=> he said it for by
Distances= [0L, 0L, 1L, 2L, 0L]