Word Prediction App

Isaac Gomes Veras

Contextualization

This Shiny app is designed to predict the next word with inputs from the Swiftkey test dataset, similar to the text prediction functions found in today's modern smartphones.

Steps taken to build a predictive model

1. A subset of the original data was sampled from the three sources (blogs, twitter and news) which is then merged into one and in-text data cleaning is performed;
2. The corresponding n-grams (Quadgram, Trigram and Bigram) are created;

3. Then term count tables are extracted from N-Grams and sorted according to frequency in descending order, these n-gram objects are saved as R-Compressed files (.RData - files).

The dataset

The prediction model built in this brilliant app came from the Swiftkey HC Corpora dataset and is made up of output from various news websites, blogs and Twitter.
The dataset contains 3 files in four languages (Russian, Finnish, German and English). This project was focused on English language datasets. The data file names are as follows:

1. en_US.blogs.txt
2. en_US.twitter.txt

App UI

N-gram based text prediction model implementing backoff algorithm. Consisting of a field for inputting a set of words, where the user must enter part of a sentence.
The output will result in the suggestion for the next terms.

The final product of the project will be an algorithm that predicts the next word in text provided with inputs from the test dataset, similar to the text prediction functions found in today's modern smartphones.

Implemented application features and algorithm

Underlying algorithm

- N-gram model with “Stupid Backoff” (Brants et al 2007)
- Checks if higher order n-gram (in this case n = 4) was seen. Otherwise, it “degrades” to a lower order model (n=3, 2); the logic could have been used even with higher requests, but ShinyApps limits the application size to 100 MB and text mining involves high computing power on large text files.

Hosted Application Link
http://i544c.shinyapps.io/Word_Prediction_App