n gram and Patricia Trie – Great Match In Natural Language Processing In Big Data at Scale

Seiya Kawashima, The University of Chicago, Radiology Department

Introduction

Although n gram was not invented by Google, n gram might have been introduced to general public by Google n gram viewer. N gram is an essential tool for Natural Language Processing such as Machine Translation, Speech Recognition, Word Prediction etc. However only generating n gram is not enough. Only generating it can be done efficiently and easily with existing tools such as Python NLTK, Natural Language Tool Kit. It must be stored in an efficient data structure so that further analysis can be as optimal as possible. Especially a project like Google n gram viewer must be at scale. To deal with huge data, generating and storing n gram must be efficient.

The key of generating n gram is to generate as many word/character patterns as possible out of data.

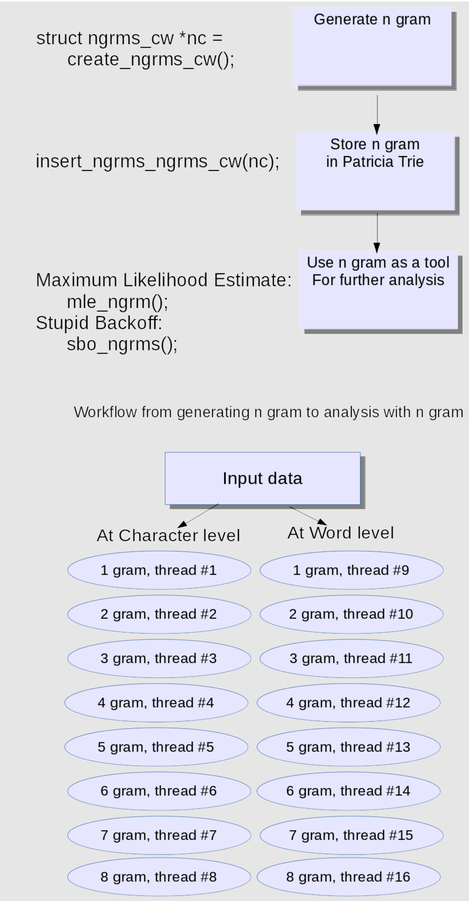
On this poster, the author introduces n gram backed by a data structure called “Patricia Trie” at scale taking advantage of multicore CPUs to challenge Google n gram viewer.

Aim

a) To challenge Google n gram viewer in terms of number of generated n gram words
b) To take advantage of multicore CPUs for huge data set.
c) To be as efficient as possible from generating n gram to further analysis with n gram.

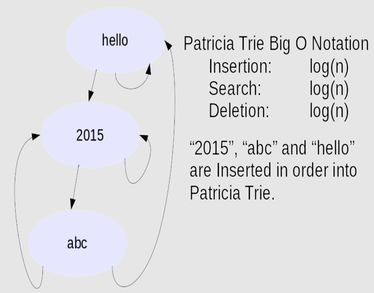
Method

a) Implement n gram using multictores from scratch.
b) Implement Patricia Trie from scratch.
c) Use Patricia Trie as a storage for n gram.
d) Gather data for n gram crawling on the Internet.
e) Run n gram on a 36 cores with 64 GB mem machine, Two Intel(R) Xeon(R) CPU E5-2630 v3 @ 2.40GHz.
f) Measure the performance as follows:
  a) # of generated n gram words per second.
  b) Total # of generated n gram words.
  c) Total # of unique words used to generate n gram words.



Workflow from generating n gram to analysis with n gram

Input Data: “Hello World”
At character level, n gram is generated as follows:
“H”,”He”,”Hel”,”Hello”, “Hello “, “Hello W”, “ Hello Wo”, …
At word level, n gram is generated as follows:
“Hello”, “Hello World”



Sample of Patricia Trie data structure

Maximum Likelihood Estimate, MLE

$$P(W_i \mid W_1 \dots W_{i-1}) = \frac{c(W_1 \dots W_i)}{c(W_1 \dots W_{i-1})}$$

Probability of W_1 followed by word up to W_{i-1})
= Frequency of word up to W_i
/
Frequency of word up to W_{i-1}

Big O Notation of Maximum Likelihood Estimate backed by Patricia Trie is log(n).
Search of Frequency of word up to W_i: log(n)
Search of Frequency of word up to W_{i-1}: log(n)

Efficient usage of Patricia Trie for n gram

Result

Total # of web pages processed	12,905
# of generated n gram words per second	3,151
Total # of generated n gram words/characters	45,608,314
# of generated n gram characters	27,357,970
# of generated n gram words	18,250,344
# of Unique words	611,593

Conclusion

There are positive and still challenging, findings on this project. Good findings are as follows:
  a) It didn't break during the experiment.
  b) It generated a great number of n gram per Second.
  c) Total # of generated n gram words/characters was huge.

Still challenging findings are as follows:
  a) Data set was not as huge as Google n gram viewer deals with.
  b) Total # of generated n gram words was not as huge as Google n gram viewer generates.

Although, with these negative findings, the Author feels confident about this project to be as good as Google n gram viewer in the near future.