Big Data Analytics on Natural Language Processing

PROJECT REPORT

Under the guidance of **Dr. US Tiwary**

Submitted By

Shubham Gupta	**IIT2013180**
Swapnil Jain	**ISM2013004**
Abhishek Jaiswal	**IIT2013129**
Shubham Chaudhary	**IIT2013149**
Chanchal Mishra	**IIT2013181**

INDIAN INSTITUTE OF INFORMATION TECHNOLOGY

ALLAHABAD

July – December, 2015

Certificate from Supervisor

This is to certify that the project work "**Big Data Analytics on Natural Language Processing**" is a bonafide work of Abhishek Jaiswal (IIT2013129), Shubham Chaudhary (IIT2013149), Shubham Gupta (IIT2013180), Chanchal Mishra (IIT2013181) and Swapnil Jain (ISM2013004) who carried out the project work under my supervision.

Date: 11th Nov 2015
Place: IIIT Allahabad

Dr. U S Tiwary
(Supervisor)

Acknowledgements

We owe special debt of gratitude to **Dr. U.S. Tiwary**, for his constant support and guidance throughout the course of our work. His sincerity, thoroughness and perseverance have been a constant source of inspiration for us. It is only because of his cognizant efforts that our project is gradually seeing the light of the day.

Abstract

- The prime focus of our work is to build a tool that allows user to carry out primitive analysis of Natural Language Processing (NLP), which could also be applied on BigData, provided the structure of the program could be modified in the said manner.

- We aim to device a platform that allows analysts, researchers, marketing experts and people of several domains to avail the benefits of NLP in the realm of Big Data.

- While we are still exploring the applications of NLP, our motive is to exploit a few of them, namely Stemming, NGram (Next-Word Prediction), Summarization and Part-Of-Speech Tagging.

- Though we have just sneaked into the field of Big Data using few hundred megabytes of data, once implemented on reasonable-sized cluster, we intend to empower it with handling of gigabytes of data within few hours.

Table of Contents

1. Our Goals

- To facilitate few applications of NLP through an interactive GUI-based tool
- To add flexibility to our tool by allowing user-defined rules
- To tap the benefits of Big Data through Hadoop Ecosystem and Map Reduce for speeding up the processing

2. Motivation

- The available contemporary programs take the form of command-line utilities. By creating a GUI, we aim to offer better control and ease of use.

- The predefined algorithms for processes like stemming or summarization used till now have hardcoded rules. We intend to add flexibility to the same by allowing user-defined rules.

- Further, our objective of employing Big Data is to improve efficiency of processing large volumes of data and to stress on use of distributed systems architecture to exploit its benefits.

3. Introduction

- Significant growth in the volume and variety of data is due to the accumulation of unstructured text data—in fact, up to 80% of all your data is unstructured text data. Companies collect massive amounts of documents, emails, social media, and other text-based information to get to know their customers better, offer customized services, or comply with federal regulations. However, most of this data is unused and untouched.

- Text analytics, through the use of natural language processing (NLP), holds the key to unlocking the business value within these vast data assets. In the era of big data, the right platform enables businesses to fully utilize their data and take advantage of the latest parallel text analytics and NLP algorithms.

- "When looking at unstructured data, for instance, we may encounter the number '31' and have no idea what that number means, whether it is the number of days in the month, the amount of dollars a stock increased over the past week, or the number of items sold today. Naked number '31' could mean anything, without the layers of context that explain who stated the data, what type of data is it, when and where it was stated, what else was going on in the world when this data was stated, and so forth. Clearly, data and knowledge are not the same thing."

- Natural Language Processing (NLP) comes into picture once we understand that Data and Knowledge are not the same thing. Data in its raw form is utterly useless unless some valuable information is extracted from it in the form of Knowledge and this is where NLP finds its role.

4. Literature Survey

- Martin Porter wrote the first successful stemming algorithm in 1980. Many implementations of Porter's algorithm that were written and freely distributed, contained subtle flaws. Hence Porter released official free software implementation in the year 2000.
- Snowball, a framework for writing stemming algorithms, was further released by Porter as an extension to his work. It was also implemented as an improved English stemmer together with stemmers for several other languages.
- In 1958, an important paper was published on Summarization, which suggested to weight the sentences of a document as a function of high frequency words, disregarding the very high frequency common words.
- In 1969, another paper described three methods for determine sentence weight. Cue method looked for presence of certain cue (indicative) words. Title method prioritized ranked sentences based on words appearing in headings. Location methods proposed that sentences occurring in beginning of paragraphs are likely to be more relevant. Researches pointed out that combination of all these methods produced best results.
- The idea of the likelihood of the next word was traced by the experiment of Claude, Shannon in Information Theory. According to him, one can drive a probability distribution for the next word, given a history of size 'n', where the probabilities of all possible next words add up to 1.
- META Group analyst Laney postulated three-fold challenges in data growth, namely, Volume, Velocity and Variety (also called the 3-V's). If these hurdles are countered, Big Data would supposedly harbour Business Intelligence by measuring things, detecting trends etc.

5. Software and Hardware Requirements

5.1 Software Requirements

- To develop this tool, the following softwares are required to be installed on the developer's PC :

  - Apache Hadoop Framework
  - Java Development Kit & JRE
  - Eclipse or NetBeans Java IDE

5.2 Hardware Requirements

- Also for setting up the multi node cluster we need 3 or more Computers with minimum configuration as follows:

  - 4 GB RAM
  - Intel Core 2 Duo or above processor

- In addition to above, we need reliable hi-speed Ethernet connectivity between all nodes and uninterrupted power supply.

6. Methodology

Our project has been divided into following segments:

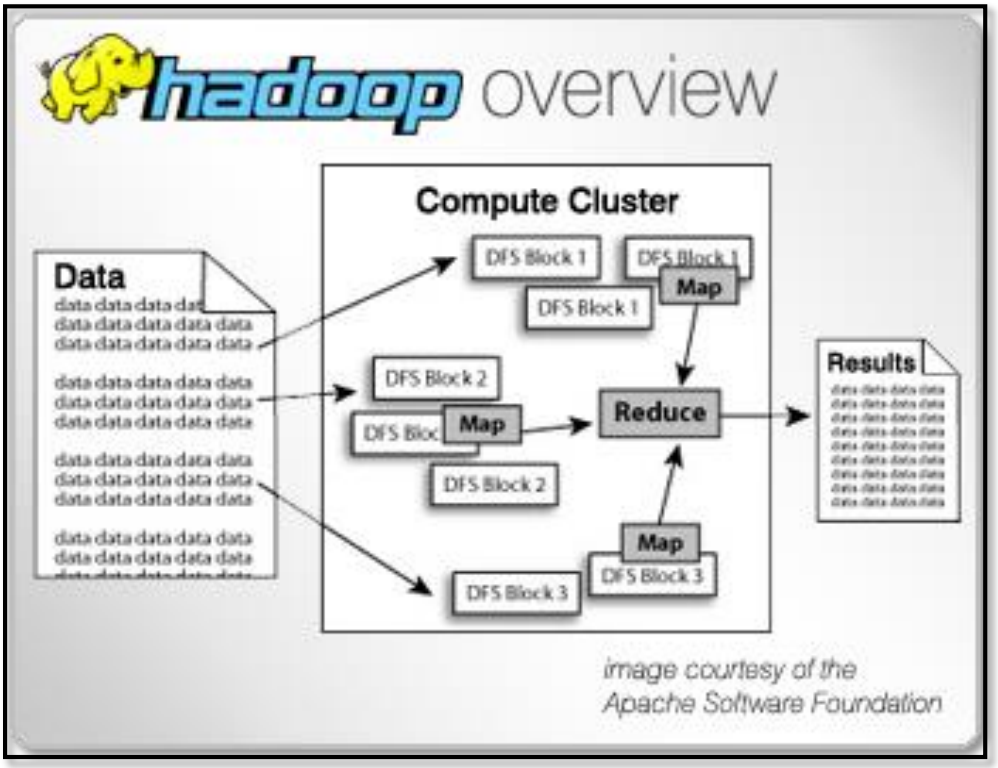
- Implementation of Stemming, N-Gram, Summarization and POS-Tagging using simple java program
- Allowing the user to define rules for Stemming, N-Gram processes
- Providing interactive GUI for each of the above processes
- Allowing different modes of inputs: text editor and input file
- Setting up Hadoop on a single node
- Running individual processes of Stemming and N-Gram on single-node
- Extending our implementation to multi-node cluster

7. Getting to know HADOOP

- Apache Hadoop is an open-source framework that allows to store and process big data in a distributed environment across clusters of computers using simple programming models. It is designed to scale up from single servers to 1000s of machines, each offering local computation and storage.

- Hadoop comprises following modules :

  - Hadoop Distributed File System (HDFS)
  - Hadoop Map Reduce
  - Hadoop YARN



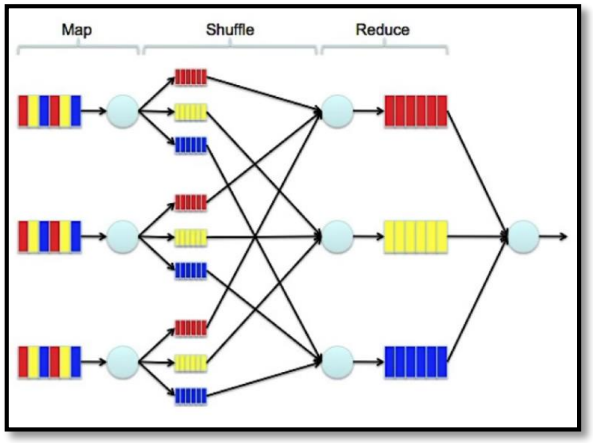
Overview of Hadoop Ecosystem

7.1 HDFS

It is a fault-tolerant Java-based distributed file system that provides scalable and reliable data storage, spanning large clusters of commodity servers. By distributing storage and computation across many servers, the combined storage resource can grow linearly with demand while remaining economical at every amount of storage. When petabytes of data spread over thousands of nodes is available in HDFS, and YARN enables multiple data access applications to process it, Hadoop users can confidently answer questions that eluded previous data platforms.

7.2 MapReduce

It splits a large data set into independent chunks and organizes them into key, value pairs for parallel processing. This parallel processing improves the speed and reliability of the cluster. Map function divides the input into ranges and creates a map task for each range. JobTracker distributes those tasks to the worker nodes. The Reduce function then collects the various results and combines them to answer the larger problem that the master node needs to solve.



Logical View of MapReduce

7.3 YARN

Yarn is the architectural centre of Hadoop that allows multiple data processing engines such as interactive SQL, real-time streaming, data science and batch processing to handle data stored in a single platform, unlocking an entirely new approach to analytics.

8. N-GRAM

- In the fields of computational linguistics and probability, an ***n*-gram** is a contiguous sequence of *n* items from a given sequence of text or speech. The items can be phonemes, syllables, letters, words or base pairs according to the application. The *n*-grams typically are collected from a text or speech corpus.

- An *n*-gram of size 1 is referred to as a "unigram"; size 2 is a "bigram" (or, less commonly, a "digram"), size 3 is a "trigram". Larger sizes are sometimes referred to by the value of *n*, e.g., "four-gram", "five-gram", and so on.

8.1 N-Gram Models

An ***n*-gram model** is a type of probabilistic language model for predicting the next item. N-gram models are now widely used in probability, communication theory, computational linguistics and data compression

8.1.1 Unigram

A unigram is an n-gram of size 1, i.e. each single word in a text document is a unigram.

8.1.2 Bigram

A bigram or digram is every sequence of two adjacent elements in a string of tokens, which are typically letters, syllables, or words; they are *n*-grams for *n*=2. The frequency distribution of bigrams in a string are commonly used for simple statistical analysis of text in many applications, including in next-word prediction, cryptography, speech recognition, and so on.

8.1.3 Trigram

These are a special case of the *n*-gram, where *n* is 3. They are often used in natural language processing for doing statistical analysis of texts.

8.2 Execution on Hadoop

Upon executing the process on Hadoop, following results were obtained

EXECUTION TIME ON HADOOP	
INPUT DATA SIZE: 250 MB	
SINGLE NODE	3 min 15 sec
TWO NODES	1 min 52 sec
INPUT DATA SIZE: 400 MB	
SINGLE NODE	4 min 43 sec
TWO NODES	3 min 2 sec
THREE NODES	1 min 57 sec

8.3 Next-Word Prediction

(An application of N-Gram)

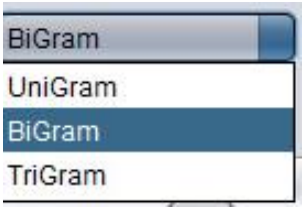
Next word prediction is process of predicting the choice of possible next words based on the previous percept history (i.e. the words fetched from the corpus).

We have implemented the word prediction system using **bigram** technique which is explained in following steps:

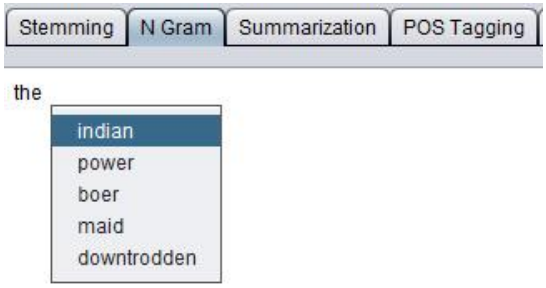
1. We take the input from the user for our **percept history base**.
2. Based on our percept history we stored the frequency of occurrence of each word after every other word in a matrix.
3. Then for each word we calculated the **top "n" words** with maximum occurrence frequency after it.
4. Now whenever user type a particular word we predict the top n words that can occur, user can select any of the suggested word which will save his effort of typing that whole word again.

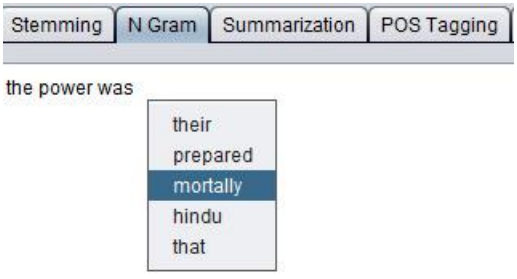
8.4 Our Implementation

1) Choose input file from the disk and select output file (not necessary if only next word prediction is to be used)
2) Choose from the options unigram, bigram, and trigram.



3) Click execute to output selected N-Gram (Uni, Bi or Tri) to file
4) Start typing on the text area, after each word you will be suggested with a list of words.





9. Stemming

- *Stemming* is a crude heuristic process that chops off the ends of words in the hope of achieving this goal correctly most of the time, and often includes the removal of derivational affixes. The goal of stemming is to reduce inflectional forms and sometimes derivationally related forms of a word to a common base form.

- Examples
  - car, cars, car's, cars' ⇒ car
  - want, wants, wanting ⇒ want

- The stem needs not to be identical to the ***morphological root*** of the word; it is usually sufficient that related words map to the same stem, even if this stem is not in itself a valid root. Many search engines treat words with the same stem as synonyms as a kind of query expansion, a process called conflation.

9.1 Porter's Algorithm

- The Porter's Algorithm for Stemming is one of the several available schemes to obtain stems of words. It must be understood that given the complexity of natural languages, none of these algorithms guarantee absolute error-free processing.

- Porter's Algorithm is a set of rules for replacement of affixes from words. These rules are applied in a pre-determined sequence, and the one having longest match is applied. Some of these rules are:

sses ⇒ ss	caresses ⇒ caress
(*v*)ing ⇒	bleeding ⇒ bleed

- As a working example, we have implemented the Porter's Algorithm in Java for demonstration of the stemming process.

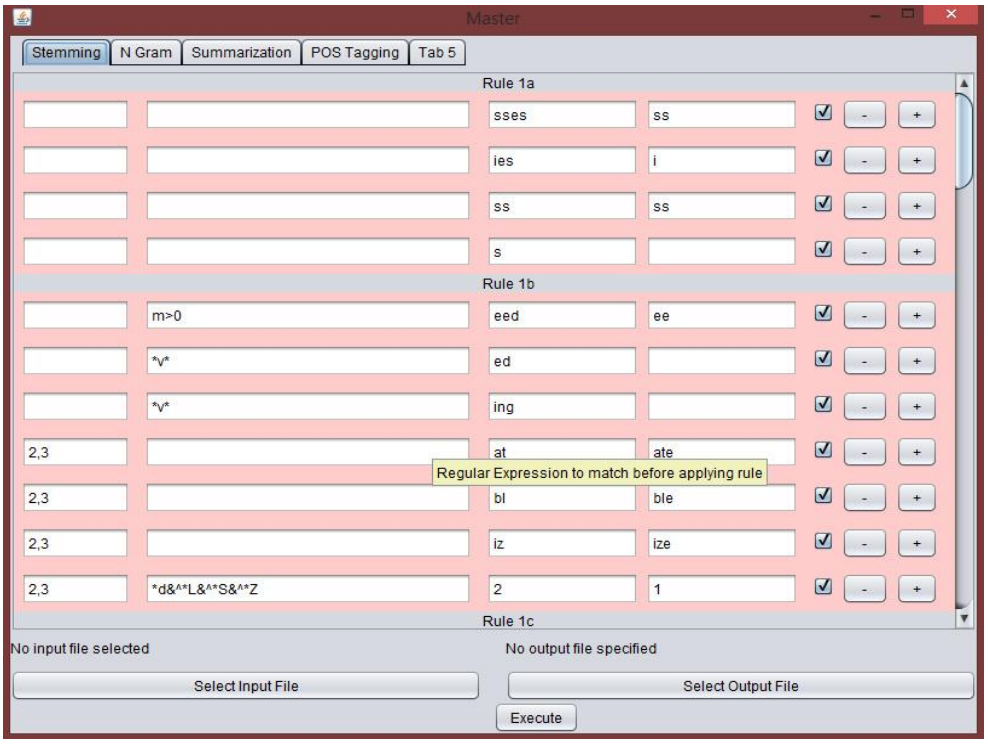
- Our aim however is to let user define the replacement rules via some easy interacting tool. With this, the task of modifying the code will be converted to keying in just the rules at any time, thereby altering rigidity of the system.

9.2 Execution on Hadoop

EXECUTION TIME ON HADOOP	
INPUT DATA SIZE: 250 MB	
SINGLE NODE	13 min 18 sec
TWO NODES	6 min 55 sec
THREE NODES	4 min 23 sec
INPUT DATA SIZE: 400 MB	
SINGLE NODE	20 min 52 sec
TWO NODES	11 min 46 sec
THREE NODES	8 min 19 sec

9.3 Our Implementation

1) Select input and output file.
2) Check or uncheck the rules according to your need.



3) You can redefine the stemming rules according to your ease by editing 3rd and 4th column values.
4) Click on execute to stem the input file.

Before stemming	After stemming
This world is full of the benefits of technology. Therefore, we are sometimes depending on technology too much. For example, one person would not move from house. He can order something to eat by telephone. He can also buy clothes by Internet. This kind of service is used delivery system, so he doesn't need to move from his house. This is one of benefits of technology called transportation.	thi world is full of the benefit of technolog therefor we ar sometim depend on technolog too much for exampl on person would not move from hous he can order someth to eat by telephon he can also bui cloth by internet thi kind of servic is us deliveri system so he doesnât need to move from hi hous thi is on of benefit of technolog call transport

10. Part-Of-Speech Tagging

- Part-of-speech tagging is used to classify words into their part-of-speech and label them according the Tag-Set which is a collection of tags. Part-of-speech tagging also known as word classes or lexical categories.

- POS tagging, also called word-category disambiguation, is the process of marking up a word in a text (corpus) as corresponding to a particular part of speech, based on both its definition, as well as its context—i.e. relationship with adjacent and related words in a phrase, sentence, or paragraph. A simplified form of this is commonly taught to school-age children, in the identification of words as nouns, verbs, adjectives, adverbs etc.

Tag	Description
CC	Coordinating conjunction
CD	Cardinal number
DT	Determiner
EX	Existential there
FW	Foreign word
IN	Preposition or subordinating conjunction
JJ	Adjective
JJR	Adjective, comparative
JJS	Adjective, superlative
LS	List item marker
MD	Modal
NN	Noun, singular or mass
NNS	Noun, plural
NNP	Proper noun, singular
NNPS	Proper noun, plural
PDT	Predeterminer
POS	Possessive ending
PRP	Personal pronoun

Tag	Description
PRP$	Possessive pronoun
RB	Adverb
RBR	Adverb, comparative
RBS	Adverb, superlative
RP	Particle
SYM	Symbol
TO	to
UH	Interjection
VB	Verb, base form
VBD	Verb, past tense
VBG	Verb, gerund or present participle
VBN	Verb, past participle
VBP	Verb, non3rd person singular present
VBZ	Verb, 3rd person singular present
WDT	Whdeterminer
WP	Whpronoun
WP$	Possessive whpronoun
WRB	Whadverb

- POS-tagging algorithms fall into two distinctive groups: rule-based and stochastic. The Tagging we've implemented is Stochastic Tagging, which assigns Tags to words based on a probability model derived from an extensive corpus of words, namely Brown Corpus, containing around 1,000,000 words along with their tags as a collection of sentences.

- Stochastic Tagging or Probabilistic Tagging, is based on Probability of certain tag being assigned to a word, given various possibilities. It requires a collection of sentences that have already been tagged (in this case, the Brown Corpus).

10.1 Approach and Formulation

1. Given a sequence of words W = w1, w2, w3 … wn, our aim is to assign sequence of tags T = t1, t2, t3 … tn

2. For this, we must find T that maximises probability P(W | T) P(T)
3. From Bayes' Rule, $P(T|W) = \alpha\, P(W|T)\, P(T)$

4. By approximation, $P(T)\, P(W|T) \approx P(t_1)P\left({}^{t_2}/_{t_1}\right) \dots P(t_{n-1})P\left({}^{t_n}/_{t_{n-1}}\right) P\left({}^{w_1}/_{t_1}\right) P\left({}^{w_2}/_{t_2}\right) \dots P\left({}^{w_n}/_{t_n}\right)$

5. With the help of Corpus, we can determine
   $P\left({}^{t_i}/_{t_{i-1}}\right) = C(t_{i-1}, t_i)/\, C\,(t_{i-1})$ and $P\left({}^{w_i}/_{t_i}\right) = C(w_i, t_i)/\, C\,(t_i)$

6. POS Tagging can be fed data via input file or the built-in text editor.
7. After providing input in either way and hitting execute, the POS-Tagged output is obtained

I'm heading to San Francisco to watch Golf Match. You are coming.	I'm PRP headingto VBG TO San NNP Francisco NNP to TO watch VB Golf NNP Match NNP You PRP are VBP coming JJ

11. Summarization

- Summary is a comprehensive and brief abstract of a long piece of text. It is obtained by recapitulation, or compendium of previously stated facts or statements.

- Automatic summarization is the process of reducing a text document with a computer program in order to create a summary that retains the most important points of the original document. As the problem of information overload has grown, and as the quantity of data has increased, so has interest in automatic summarization. Technologies that can make a coherent summary take into account variables such as length, writing style and syntax. Automatic data summarization is a very important area within machine learning and data mining. An example of the use of summarization technology is search engines such as Google.

- The main idea of summarization is to find a representative subset of the data, which contains the information of the entire set. Document summarization, tries to automatically create a representative summary or abstract of the entire document, by finding the most informative sentences.

11.1 Sentence Score Determination

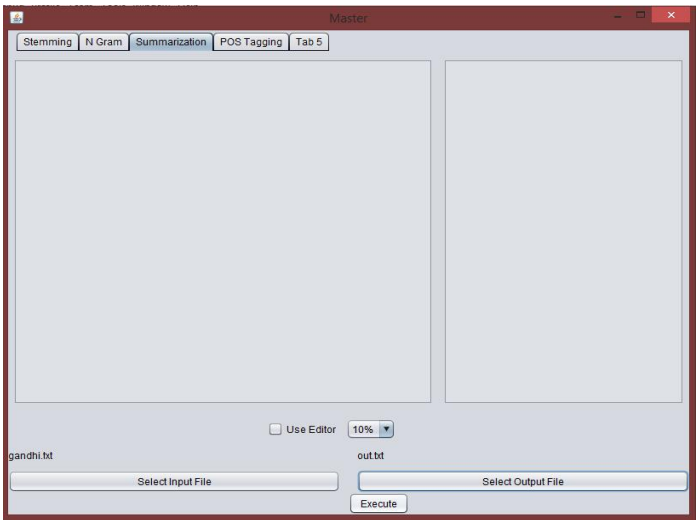
- We implemented the auto summarization algorithm, by breaking the large text into sentences.
- Then we created an Intersection function which receives two sentences, and returns a score for the ***intersection*** between them. We just split each sentence into words/tokens, count how many common tokens we have, and then we normalize the result with the average length of the two sentences.

$$f(S1, S2) = \frac{M \times \sqrt{W_1} \times \sqrt{W_2}}{W_1 + W_2}$$

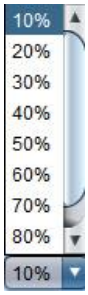
- In fact, we just converted our text into a fully-connected weighted graph! Each sentence is a node in the graph and the two-dimensional array holds the weight of each edge!
- We calculate the score for each sentence by just summing up its mutual scores with the other sentences, score is a metric to measure the importance of a particular sentence.
- Then we let user select the percentage of text to be produced as summary
- Finally, every 1 out of n consecutive sentences is picked, where n is determined by total no of sentences and percentage of summary needed.

11.2 Our Implementation

1) Select Input and Output file by clicking on the buttons and choosing files from the disk or you can use the Editor to simply paste your text in left text area.



2) Choose the percentage from the drop box which is a measure to limit the word count of resulting summary.



3) We have chosen 10% as the summary and have taken a paragraph to be summarised.

Input	Output
Children, there is not a single country in the whole world where the name of Mahatma Gandhi is not known. Do you know why Gandhiji became so famous? It was because h e dedicated his whole life to the service of the motherland, and service of humanity. Tod ay, I am going to tell you in brief, the story of Mahatma Gandhi, the father of the Nation, o r Bapuji, as he is affectionately called. In the early days our country was made up of a la rge number of small Princely Kingdoms. Porbandar in Gujarat was one such Princely K ingdom. Gandhiji's father Karamchand Gandhi.	Today I am going to tell you in brief the story of M ahatma Gandhi the father of the Nation or Bapuji as he is affectionately called.

12. Timeline

12.1 Pre Mid-SEM

- In the initial days, we read about Natural Language Processing from video tutorial series of MITOpenCourseware
- We implemented the Porter's Algorithm in java
- We learnt about Big Data and Hadoop through Lynn Langit's video tutorial series

12.2 Post Mid-SEM

- We modified the Porter's algorithm to support user-defined rules entered via GUI
- We implemented the Next-Word Prediction via N-Gram scheme and built a popup-suggestion based editor using it
- We implemented Summarization process by employing the sentence score method
- We implemented the POS-Tagger Stochastic Probabilistic Model for English Language based on Brown's Corpus
- We wrote the MapReduce codes for Stemming and N-Gram
- The MapReduce codes were run on single node as well as multi-node cluster and their run-time were recorded

13. References

- **[**7.2 **MapReduce]** - J Dean and S Ghemawat, "MapReduce: Simplified Data Processing on Large Clusters", Google Inc., 2004

- **[**8.1 **N-Gram Models]** - Cornell University, N-Gram Models, a document on next-word prediction technique by employing N-Gram

- **[**8.3 **Next-Word Prediction]** – Next Word Prediction, a document describing approach to predict next word, by Department of Computer Science, Cornell University

  https://www.cs.cornell.edu/courses/CS4740/2012sp/lectures/smoothing+backoff-1-4pp.pdf

- **[**9 **Stemming]** - A.G. Jivani, "A Comprehensive Study of Stemming Algorithms", University of Baroda, 2011

- **[**9.1 **Porter's Algorithm]** - The Porter Stemming Algorithm, a comprehensive description of the algorithm by Martin Porter himself. Retrieved from:

  http://tartarus.org/~martin/PorterStemmer/

- **[**10 **Part-Of-Speech Tagging]** – A document describing approaches to POS-Tagging

  http://www.stat.columbia.edu/~madigan/DM08/hmm.pdf

- **[**10.1 **POS-Tagging]** - University of Maryland, Part-Of-Speech Tagging, a document by Department of Computer Science on POS Tagging

- **[**11 **Summarization]** - Y.S. Reddy and A.P. Siva Kumar, "An Efficient Approach for Web document summarization by Sentence Ranking", International Journal of Advanced Research in Computer Science and Software Engineering, 2012

14. Suggestions by Board Members