**TOPIC MODELING USING BIG DATA ANALYTICS**

A PROJECT REPORT

Submitted in the partial fulfilment of the requirements

for the award of the degree of

**BACHELOR OF TECHNOLOGY**

**IN**

**COMPUTER ENGINEERING**



**Under The Supervision of**  **Submitted by**

Dr. Tanvir Ahmad Farheen Nilofer(12-CSS-21)

Head & Professor Sarah Masud(12-CSS-57)

**DEPARTMENT OF COMPUTER ENGINEERING**

**FACULTY OF ENGINEERING AND TECHNOLOGY**

**JAMIA MILLIA ISLAMIA , NEW DELHI - 110025**

**(Year - 2015)**

**TABLE OF CONTENTS**

CERTIFICATE

ACKNOWLEDGEMENT

ABSTRACT

CHAPTER 1

INTRODUCTION

BIG DATA

HADOOP

HADOOP STRUCTURE

CONFIGURATION

MAPREDUCE

CHAPTER 2

LITERATURE REVIEW

KNOWLEDGE REPRESENTATION TECHNIQUES

TEXT MINING TECHNIQUES

MASSIVE MINING TOOLS

CHAPTER 3

TERMINOLOGY

METHODOLOGY

ALGORITHM

FLOWCHART

RESULTS

CONCLUSION

FUTURE WORK

REFERENCES

**LIST OF FIGURES**

Figure 1. Components of Hadoop --9

Figure 2. Components of MapReduce --11

Figure 3. Tasks of Map and Reduce --11

Figure 4. Three System Cluster --14

Figure 5. Namenode formatted --14

Figure 6. Starting Hadoop --15

Figure 7. All process running --15

Figure 8. Copy files from local to hdfs --15

Figure 9. Running jar of word count data --16

Figure 10. MapReduce Complete --16

Figure 11. Word Count output --19

Figure 12. Typical Application of Text Mining for Ontology Extraction --25

Figure 13. Noun Extraction From Tagged List --26

Figure 14. Formula to calculate Similarity Structure --26

Figure 15. Formula to calculate Degree of specificity --27

Figure 16. Relevance Score Calculation --27

Figure 17. Text Window Frequency Vector of Attributes --35

Figure 18. Mutual Information among Attributes --36

Figure 19. Pruned Concepts --37

Figure 20. Mapping of Concepts and subconcepts --38

**CERTIFICATE**

This is to certify that the dissertation/project report (Course Code) entitled “Topic Modeling Using Big Data Analytics”, is an authentic work carried out by **Miss Farheen Nilofer** and **Miss Sarah Masud**.

The work is submitted in partial fulfilment of the requirements for the award of the degree of Bachelor of Technology in Computer Engineering under my guidance. The matter embodied in this project work has not been submitted earlier for the award of any degree or diploma to the best of my knowledge and belief.

**Date: 21/12/2015**

**Dr. Tanvir Ahmad,**

Professor,

Dept. of Computer Engg.

Jamia Millia Islamia.

**ACKNOWLEDGEMENT**

We would like to thank our mentors Dr. Tanvir Ahmad (Head of Department, Dept. of Computer Engineering) and Mr. Rafeeq Ahmed (Ph.D scholar, Dept. of Computer Engineering) for giving us the opportunity to undertake the project. We thank them for their immense guidance, and appreciate their timely engagement.

We would like to extend special gratitude to the the assistants and lab coordinators of the Department for providing us the infrastructural facilities necessary to sustain the project.

**ABSTRACT**

Topic models are a suite of algorithms that uncover the hidden thematic structure in document collections. These algorithms help us develop new ways to search, browse and summarize large archives of texts.The structure uncovered by topic models can be used to explore an otherwise unorganized collection. The models used for Topic modelling are largely computation dependent. This is where the techniques of Big Data come into use. Big data processing environments such as Hadoop are useful for the analysis of large data sets, best designed to handle unstructured data.

**CHAPTER 1**

**INTRODUCTION**

**Problem Statement: Topic Modelling using the techniques of Big Data.**

**Topic Modeling:**

Topic models provide a simple way to analyze large volumes of unlabeled text. A "topic" consists of a cluster of words that frequently occur together. Using contextual clues, topic models can connect words with similar meanings and distinguish between uses of words with multiple meanings.

**Big Data:**

**VOLUME->VARIETY->VELOCITY->VARIABILITY**

Big data is being generated by everything around us at all times. Every digital process and social media exchange produces it. Systems, sensors and mobile devices transmit it. Big data is arriving from multiple sources at an alarming velocity, volume and variety. To extract meaningful value from big data, you need optimal processing power, analytics capabilities and skills.

Big data is changing the way people within organizations work together. It is creating a culture in which business and IT leaders must join forces to realize value from all data. Insights from big data can enable all employees to make better decisions—deepening customer engagement, optimizing operations, preventing threats and fraud, adaptive learning and capitalizing on new sources of revenue. But escalating demand for insights requires a fundamentally new approach to architecture, tools and practice.

**Hadoop:**

Hadoop is an Apache open source framework written in java that allows distributed processing of large datasets across clusters of computers using simple programming models. A Hadoop frame-worked application works in an environment that provides distributed storage and computation across clusters of computers. Hadoop is designed to scale up from single server to thousands of machines, each offering local computation and storage.

## **Hadoop Architecture:** Hadoop framework includes following four modules:

1. **Hadoop Common:** These are Java libraries and utilities required by other Hadoop modules. These libraries provides filesystem and OS level abstractions and contains the necessary Java files and scripts required to start Hadoop.
2. **Hadoop YARN:** This is a framework for job scheduling and cluster resource management.
3. **Hadoop Distributed File System (HDFS):** A distributed file system that provides high-throughput access to application data.
4. **Hadoop MapReduce:** This is YARN-based system for parallel processing of large data sets.

Since 2012, the term "Hadoop" often refers not just to the base modules mentioned above but also to the collection of additional software packages that can be installed on top of or alongside Hadoop, such as Apache Pig, Apache Hive, Apache HBase, Apache Spark etc.

## **Hadoop Distributed File System :**

Hadoop can work directly with any mountable distributed file system such as Local FS, HFTP FS, S3 FS, and others, but the most common file system used by Hadoop is the Hadoop Distributed File System (HDFS). The Hadoop Distributed File System (HDFS) is based on the Google File System (GFS) and provides a distributed file system that is designed to run on large clusters (thousands of computers) of small computer machines in a reliable, fault-tolerant manner.

HDFS uses a **master/slave architecture** where master consists of a single **NameNode,** that manages the file system metadata and one or more slave **DataNodes,** that store the actual data.

A file in an HDFS namespace is split into several blocks and those blocks are stored in a set of DataNodes. The NameNode determines the mapping of blocks to the DataNodes. The DataNodes takes care of read and write operation with the file system. They also take care of block creation, deletion and replication based on instruction given by NameNode. We can use following diagram to depict these four components available in Hadoop framework.

## Hadoop Architecture

## **Fig 1. Components of Hadoop**

## 

## **MapReduce:**

Hadoop MapReduce is a software framework for easily writing applications which process big amounts of data in-parallel on large clusters (thousands of nodes) of commodity hardware in a reliable, fault-tolerant manner.

The term MapReduce actually refers to the following two different tasks that Hadoop programs perform:

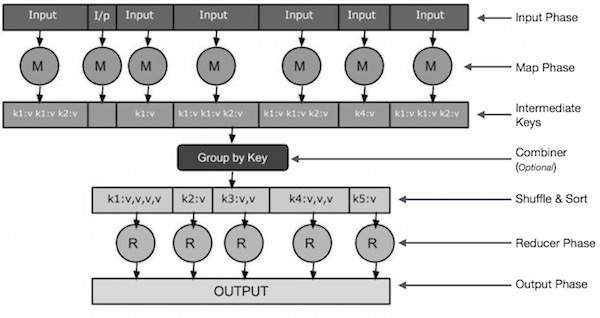
(i)The Map Task: This is the first task, which takes input data and converts it into a set of data, where individual elements are broken down into tuples (key/value pairs).

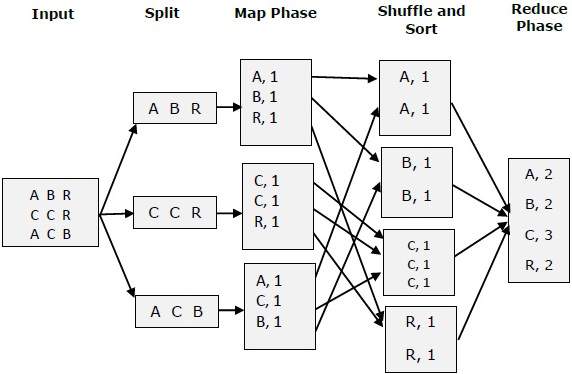
(ii)The Reduce Task: This task takes the output from a map task as input and combines those data tuples into a smaller set of tuples. The reduce task is always performed after the map task.

Typically both the input and the output are stored in a file-system. The framework takes care of scheduling tasks, monitoring them and re-executes the failed tasks.

The MapReduce framework consists of a single master JobTracker and one slave TaskTracker per cluster-node. The master is responsible for resource management, tracking resource consumption/availability and scheduling the jobs component tasks on the slaves, monitoring them and re-executing the failed tasks. The slaves TaskTracker execute the tasks as directed by the master and provide task-status information to the master periodically.

The JobTracker is a single point of failure for the Hadoop MapReduce service which means if JobTracker goes down, all running jobs are halted.

**Fig 2. Components of MapReduce.**



**Fig 3. Tasks of Map and Reduce**

**CONFIGURATION OF HADOOP:**

**Version Used: Hadoop 1.2.1**



**Fig 4. Three System Cluster**

**Hadoop Configuration Steps:**

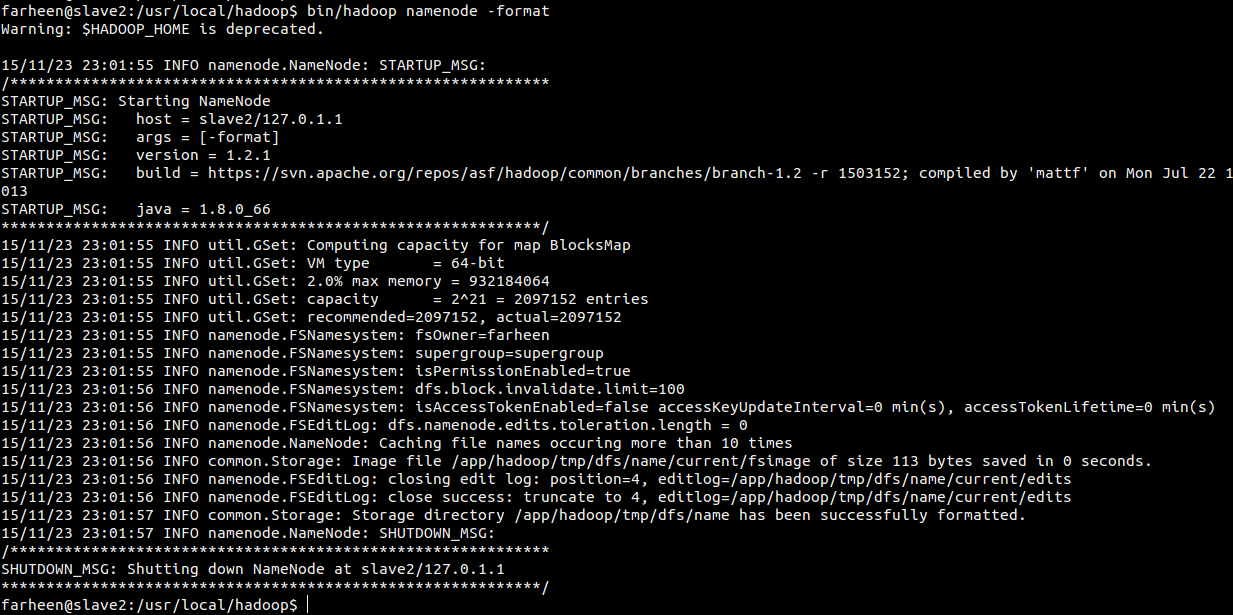
1. **INSTALL JAVA**
   1. sudo apt-get install sun-java-8-jdk
2. **CONFIGURE SSH**
   1. ssh-keygen -t rsa -P “ “
   2. ssh localhost
   3. ssh slave
3. **HADOOP INSTALL**
   1. Download Hadoop
   2. cd /usr/local
   3. sudo tar hadoop
   4. sudo chown -R hduser:hadoop hadoop
4. **ADD THE FOLLOWING PROPERTIES IN** conf/core-site.xml
   1. hadoop.tmp.dir
   2. fs.default.name
5. **ADD THE FOLLOWING PROPERTIES IN** conf/mapred-site.xml
   1. mapred.job.tracker
6. **ADD THE FOLLOWING PROPERTY IN** hdfs-site.xml
   1. dfs.replication
7. **CONFIGURE** /etc/hosts

**Commands To Run Hadoop:**

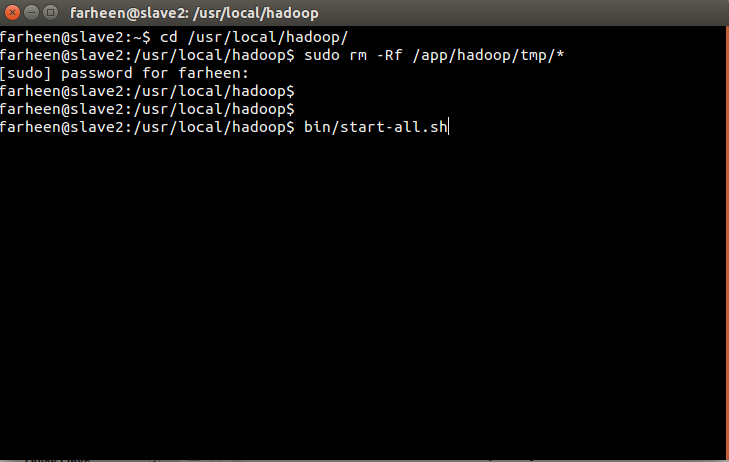
1. /usr/local/hadoop/bin/hadoop namenode-format
2. /usr/local/hadoop/bin/hadoop/bin/start-all.sh
3. /usr/local/hadoop/bin/hadoop dfs -copyFromLocal <source> <destination>
4. /usr/local/hadoop/bin/hadoop/bin/stop-all.sh

**Word Frequency Program : Processing**

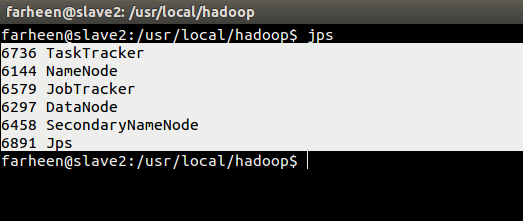
**Fig 5. Namenode formatted**





**Fig 6. Starting Hadoop**   

**Fig 7. All process running**





**Fig 8. Copy files from local to hdfs**

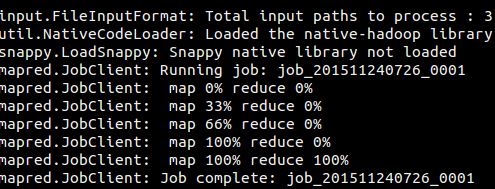
copy.png

**Fig 9. Running jar of word count data**

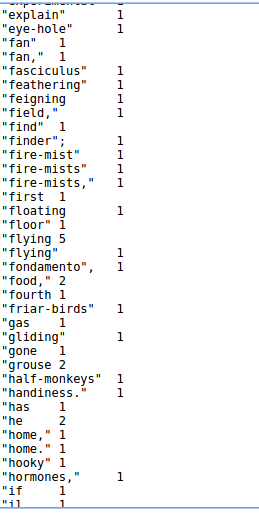
jar.png



**Fig 10. Map Reduce Complete**



**Fig 11. Word Count output:**



**CHAPTER 2**

**LITERATURE REVIEW**

**Knowledge Representation:**

Knowledge representation and reasoning (KR) is the field of artificial intelligence (AI) dedicated to representing information about the world in a form that a computer system can utilize to solve complex tasks such as diagnosing a medical condition or having a dialog in a natural language. Knowledge representation incorporates findings from psychology about how humans solve problems and represent knowledge in order to design formalisms that will make complex systems easier to design and build. Knowledge representation and reasoning also incorporates findings from logic to automate various kinds of reasoning, such as the application of rules or the relations of sets and subsets.

Examples of knowledge representation formalisms include semantic nets, systems architecture, Frames, Rules, and ontologies.

**Knowledge Representation Techniques:**

All of these, in different ways, involve hierarchical representation of data.

* Lists - linked lists are used to represent hierarchical knowledge
* Trees - graphs which represent hierarchical knowledge. LISP, the main programming language of AI, was developed to process lists and trees.
* Semantic networks - nodes and links - stored as propositions.
* Frames - Describe objects. Consist of a cluster of nodes and links manipulated as a whole. Knowledge is organised in slots. Frames are hierarchically organised. Example on p. 151 of Stillings
* Scripts - Describe event rather than objects. Consist of stereotypically ordered causal or temporal chain of events.
* Rule-based representations (Newell and Simon) - used in specific problem-solving contexts. Involve production rules containing if-then or situation-action pairs. Specific example: problem space representations.

Contain:

Initial state

Goal state

Legal operators, i.e. things you are allowed to do

* Logic-based representations - may use deductive or inductive reasoning. Contain:

Facts and premises

Rules of propositional logic (Boolean - dealing with complete statements

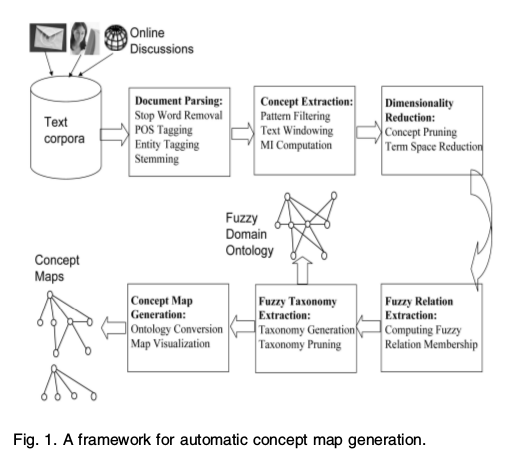
Rules of predicate calculus (allows use of additional information about objects in the proposition, use of variables and functions of variables

* Measures of certainty - may involve Certainty Factors (eg. If symptom then (CF) diagnosis) which could be derived from expert estimation or from statistical data; Bayesian probability; or Fuzzy logic (in which the concepts or information itself has some associated certainty value).
* Ontology: Formal representations of a set of concepts within a domain and the relationships between those concepts.In computer science, ontology is a formal naming and definition of objects and the inter-relation between them. It is the representation of entities and objects and events, along with their properties and relations.
* Domain Ontology:A type of ontology which is specific to domain, the meaning of the word forming the concepts changes when moving from one system to another. Eg: The word cards has many different meanings. In context of poker it represents a deck of card, in context of birthdays, a greeting card, while in domain for computer , it might represent a punch card or a graphics card.

**Text Mining:**

Text mining, also referred to as text data mining, refers to the process of deriving high-quality information from text. High-quality information is typically derived through the devising of patterns and trends through means such as statistical pattern learning. Text mining usually involves the process of structuring the input text (usually parsing, along with the addition of some derived linguistic features and the removal of others, and subsequent insertion into a database), deriving patterns within the structured data, and finally evaluation and interpretation of the output. 'High quality' in text mining usually refers to some combination of relevance, novelty, and interestingness.

Text analysis involves information retrieval, lexical analysis to study word frequency distributions, pattern recognition, tagging/annotation, information extraction, data mining techniques including link and association analysis, visualization, and predictive analytics.



**Fig 12. Typical Application of Text Mining for Ontology Extraction**

**Some Text Analysis Processes:**

**Information retrieva**l or identification of a corpus is a preparatory step: collecting or identifying a set of textual materials, on the Web or held in a file system, database, or content corpus manager, for analysis. Although some text analytics systems apply exclusively advanced statistical methods, many others apply more extensive natural language processing, such as part of speech tagging, syntactic parsing, and other types of linguistic analysis.

**Named entity recognition** is the use of gazetteers or statistical techniques to identify named text features: people, organizations, place names, stock ticker symbols, certain abbreviations, and so on. Disambiguation — the use of contextual clues — may be required to decide where, for instance, "Ford" can refer to a former U.S. president, a vehicle manufacturer, a movie star, a river crossing, or some other entity.

**Recognition of Pattern Identified Entities**: Features such as telephone numbers, e-mail addresses, quantities (with units) can be discerned via regular expression or other pattern matches.

**Relationship, fact, and event Extraction:** identification of associations among entities and other information in text.

**Sentiment analysis** involves discerning subjective (as opposed to factual) material and extracting various forms of attitudinal information: sentiment, opinion, mood, and emotion. Text analytics techniques are helpful in analyzing, sentiment at the entity, concept, or topic level and in distinguishing opinion holder and opinion object.

**Quantitative text analysis** is a set of techniques stemming from the social sciences where either a human judge or a computer extracts semantic or grammatical relationships between words in order to find out the meaning or stylistic patterns of, usually, a casual personal text for the purpose of psychological profiling etc.

**Real World Applications of Hadoop:**

**1. Analyze life-threatening risks:**

Suppose you’re a doctor in a busy hospital. How can you quickly identify patients with the biggest risks? How can you ensure that you’re treating those with life-threatening issues, before spending your time on minor problems?

**2. Identify warning signs of security breaches:**

What if you could stop security breaches before they happened? What if you could identify suspicious employee activity before they took action? The solution lies in data, with the ability to now mine and correlate people, business, and machine-generated data all in one seamless analytics environment, we can get a far more complete picture of who is doing what and when.

**3. Prevent hardware failure:**

Machines generate a wealth of information–much of which goes unused. Once you start collecting that data with Hadoop, you’ll learn just how useful this data can be.For instance, this recent webinar on “Practical Uses of Hadoop,” explores one great example. Capturing data from HVAC systems helps a business identify potential problems with products and locations.

**4. Understand what people think about your company**

Do you ever wonder what customers and prospects say about your company? Is it good or bad? Just imagine how useful that data could be if you captured it.With Hadoop, you can mine social media conversations and figure out what people think of you and your competition. You can then analyze this data and make real-time decisions to improve user perception.

**5. Hadoop in bioinformatics**

Because of its ability to store and process complex data of almost any kind, Hadoop provides a platform that makes it easier to integrate and analyze not just nucleotide sequences, but also PubMed articles, X-ray crystallography showing molecular structure and other highly valuable laboratory data and analyses. Combining diverse scientific data on Hadoop provides a huge opportunity for new approaches to understanding molecular function in gene activation and disease pathways.

**Massive Mining Techniques:**

The Big Data phenomenon is intrinsically related to the open source software revolution. Large companies such as Facebook, Yahoo!, Twitter, LinkedIn benefit and contribute to open source projects. Big Data infrastructure deals with Hadoop, and other related software as:

**Apache Hadoop** : software for data-intensive distributed applications, based in the MapReduce programming model and a distributed file system called Hadoop Distributed Filesystem (HDFS). Hadoop allows writing applications that rapidly process large amounts of data in parallel on large clusters of compute nodes. A MapReduce job divides the input dataset into independent subsets that are processed by map tasks in parallel. This step of mapping is then followed by a step of reducing tasks. These reduce tasks use the output of the maps to obtain the final result of the job.

**Apache Hadoop related projects: Apache Pig, Apache Hive, Apache HBase, Apache ZooKeeper, Apache Cassandra, Cascading, Scribe and many others.**

**Apache S4 and Strorm :** platform for processing continuous data streams. S4 is designed specifically for managing data streams. S4 apps are designed combining streams and processing elements in real time.

In Big Data Mining, there are many open source initiatives. The most popular are the following:

**Apache Mahout:** Scalable machine learning and data mining open source software based mainly in Hadoop. It has implementations of a wide range of machine learning and data mining algorithms: clustering, classification, collaborative filtering and frequent pattern mining.

**R:** open source programming language and software environment designed for statistical computing and visualization. R was designed by Ross Ihaka and Robert Gentleman at the University of Auckland, New Zealand beginning in 1993 and is used for statistical analysis of very large data sets.

**Pegasus**: big graph mining system built on top of MapReduce. It allows to find patterns and anomalies in massive real-world graphs.

**CHAPTER 3:**

**TERMINOLOGY:**

**Fuzzy Ontology:**

A fuzzy ontology is a 6 tuple set Ont=<X,A,C,Rxc,RAC,RCC>, where X is a set of objects, A is a set of attributes describing the objects, and C is a set of concepts. the fuzzy relation Rxc, assigns a membership to the pair(xi,ci), the fuzzy relation RAC , defines the mapping from the set of attributes A to the set of concepts C, and the fuzzy relation Rcc ,defines the strength of the subclass among the set of concepts in C.

**Fuzzy Set:**

A fuzzy set F consists of a set of objects drawn from a domain X and the membership of each object xi  in F defined by a membership function, with values between [0,1].

**Fuzzy Relation:**

A fuzzy relation Rxy is defined as the fuzzy set R on domain X x Y, where X and Y are two crisp sets.

**Fuzzy subsumption:**

If the membership of every attribute ai ,in concept Cy is greater than or equal to the threshold, and the membership of the corresponding attributes in Cx is also greater than or equal to the threshold, then the concept Cx is the subconcept of Cy .

**METHODOLOGY:**

**PHASE 1:**

Setting up hadoop multi cluster involves setting up hadoop in pseudo distributed mode, and after successful implementation scaling it to milti nodes. The steps involved are as follows:

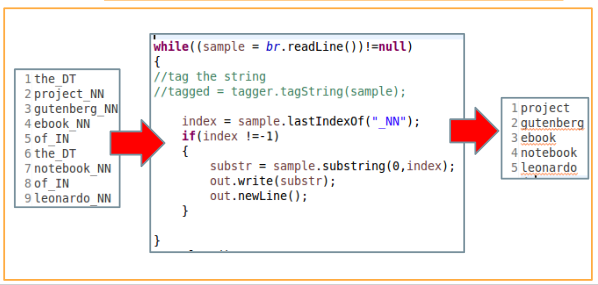
1. Installation of Java: Hadoop framework is implemented in Java, and requires JDK to compile and run its program. Hence we need to set up Java before setting up hadoop.
2. Configure the SSH: We install the openssh client and server to provide an encrypted mode of communication among the nodes. In order to prevent the human intervention we set a password less ssh connection for the same.
3. Installing Hadoop: This step involves downloading and unzipping the required version of hadoop under the usr/local directory and then assigning desired group ownership and user permissions.
4. Setting up temp directory:Hadoop needs a directory where it stores the data files, port listing, and intermediate data. Hadoop’s default configuration uses hadoop.temp.dir as the base directory for both local file system and HDFS.
5. In the core-site.xml add the property to locate the base folder, also specify the default file system, along with the host and port. In mapred-site,xml specify the host and port where the job racker will be running. In the hdfs-site.xml specify the number of replications for a file when it will be created.
6. Make changes in the slaves and masters files, to specify the list of master(Namenode) and slaves(Datanodes).
7. Make necessary changes in the /etc/hosts file so that all the nodes within the cluster are aware of all other nodes in the clusters, and are able to ping them.
8. Test the system against some standard examples of wordcount and PI value calculation.

**PHASE II:**

Major Steps and techniques, in Fuzzy Domain Ontology Extraction.

**Standard preprocessing:**

1. **Stop Word removal**: In computing, stopwords usually refers to the most common words in a language, there is no single universal list of stopwords, and is build upon as per the domain of work.
2. **Tokenization**: The process of breaking a stream of text into words, phrases, symbols or other elements called tokens. The list of tokens become input for future parsing.
3. **Tagging:** The process of classifying words into their parts of speech and labeling them accordingly is known as part-of-speech tagging.
4. **Noun Extraction:** The tagged output is a tuple consisting of the words and its POS in the form of <word\_tag>. We need to extract only those words that are tagged as noun. This is because only nouns can be used to derive concepts. The nouns represent entities while other POSs represent some work done upon the entity or add to the description of entity, not the entity itself. Since ontology deals with entity, we need only those whose words, i.e nouns.



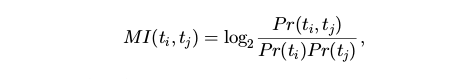
**Fig 13. Noun Extraction From Tagged List.**

**Text Windowing**:

A moving word of 5 to 10 words is taken, to find the frequency of occurrence between the terms in the window (target word with all other words within the window). A window is a sequential collection of 5 to 10 words. Within each window, the statistical information among tokens is collected to develop collocational expressions.The result of text windowing is the context vector.

**Calculation Mutual Information:**

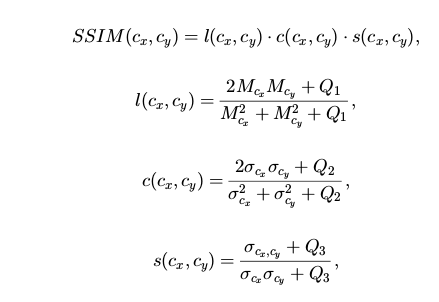
It is an information theoretic method to compute the dependency between entities and is defined as joint probability that both terms appear in a text window, upon the individual probability of the term in the text window.



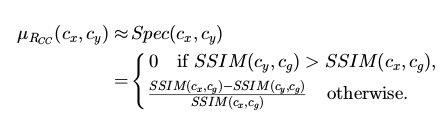
**Fig 13. Formula to calculate Mutual Information**

**Concept Filtering:**

If the Mutual information provided by an attribute falls above the threshold value then it will be treated as a potential concept, otherwise the relative importance of the concept is low.



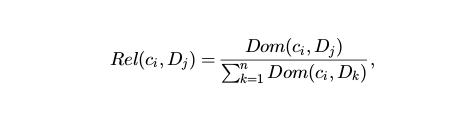
**Fig 14. Formula to calculate Similarity Structure**



**Fig 15. Formula to calculate Degree of specificity.**

**Relevance Scoring:**

To further filter the noisy concepts, we adopt a concept similar to TF IDF. If a concept is significant for a particular domain, it will appear more frequently in that domain when compared with its appearance in other domains. The relevance score of a concept is the ratio of number of documents containing the concept divided by the total number of documents in the corpus. The higher the value of the score, the more relevant is the concept.



**Fig 16. Relevance Score Calculation**

**Fuzzy Relation Extraction:**

The last few steps toward our ontology extraction method is fuzzy taxonomy generation based on the subsumption relations among the extracted concepts. The degree of subsumption (specificity) of Cx to Cy is based on the ratio of the sum of the minimal membership values of the common terms belonging to both concepts to the sum of the membership values of terms in concept CX. An improvement over this method is the implementation of SSIM to text vectors. Any two concepts Cx and Cy  could be said to be similar of their structural similarity is high. To formulate our specificity, we first compute the common concept Cg as fuzzy intersection of concepts Cx and Cy. Then we determine the direction of the specialization relation.

**Fuzzy Taxonomy Extraction:**

Before building the fuzzy taxonomy, we only select the subsumption relations if specificity from concept X to concept Y is greater than a threshold value (empirically calculated), and also greater than the specificity from concept Y to concept X.

**Time Complexity:**

The general computational complexity of the algorithm is characterised by *O(k2n+n2k)*, where K is the number of attributes , and n is the cardinality of the pruned concepts C. For large datasets with k==n, the computational complexity is expressed as upper bound ***O(n3)***. Thus, by controlling the filtering and pruning threshold, we can reduce the concept spacialy. Also, one can apply dimensionality reduction techniques to reduce the dimension of K to represent, the reduced dimensionality of term space instead.

**Algorithm:**

**Input**: Set of three documents

**Output:**Table Concepts and their subconcepts

**Procedure:**

**1.** For each document d ϵ D

1.1 Select the term ti

1.2 Check for stop word

1.3 Part Of Speech Tagging

1.4 Noun Extraction(\_NN only)

**2.** Unique Word Extraction from the corpus.

**3.** Create list mylist->unique attribute list.

**4**. Create list mylist2->all attributes list.

**5.** Calculate frequency of words that occur together in a window by Attribute-Attribute Matrix AA

AA[len1][len1]; len1 -> lenght of mylist

AA[attribute1][attribute2]] += 1 ; attribute(i) ϵ mylist2

**6**. Mutual Information Calculation for Concept Filtering

**6.1**

If *temp > threshold* then

else ; where prtij-> arr[][]/len, prti = count[i]/len

**6.2** Concept Pruning

If Concept Count >threshold, store potential concepts *concept\_count[]*

**7**. Attribute Concept calculation (AC)

If <attribute> ϵ *concept\_count[]* then  *AC[i][j] = prob[i][j]*

**8.** Relevance score calculation using *tf-idf*

**9** . Concept Concept(CC) Calculation : Fuzzy Relation Extraction

9.1 For each pair of concepts *(Ci,Cj)* ϵ AC[i][j] , compute taxonomy relation using *spec(Ci,Cj).*

*9.2* If *spec(Ci,Cj) > threshold then*

*CC[][] = CC[][] U (Ci,Cj)*

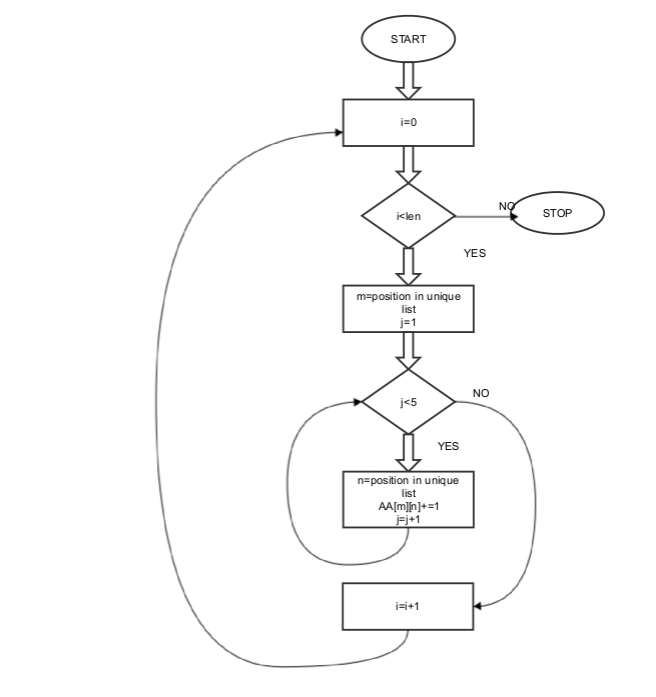
**10**. Taxonomy Extraction

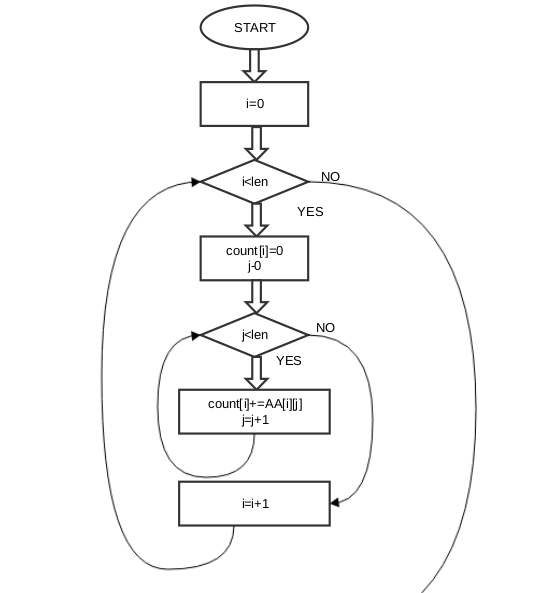
If *CC[i][j] < CC[j][i]* then *CC[i][j]=0*

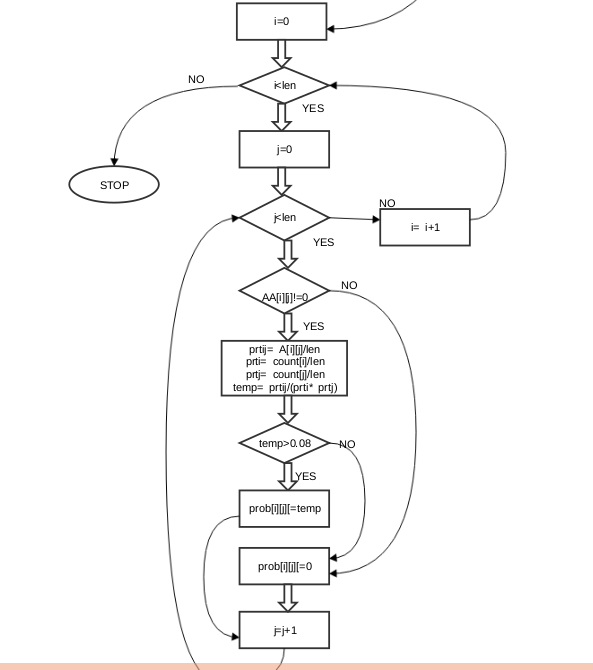
**11**. Removal of weaker concepts.

**FLOWCHARTS**

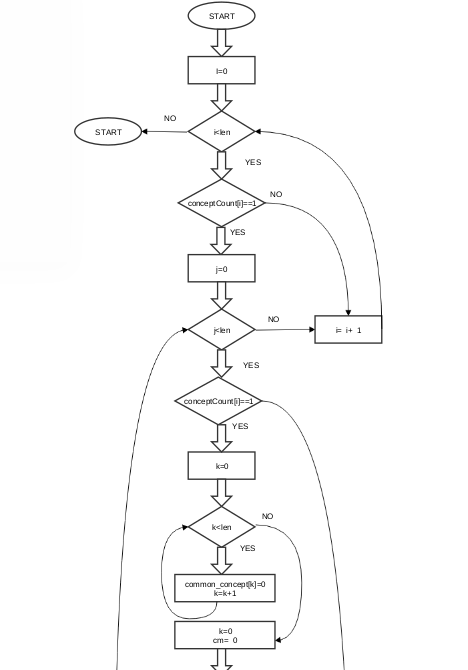
1. **Context Vectors**

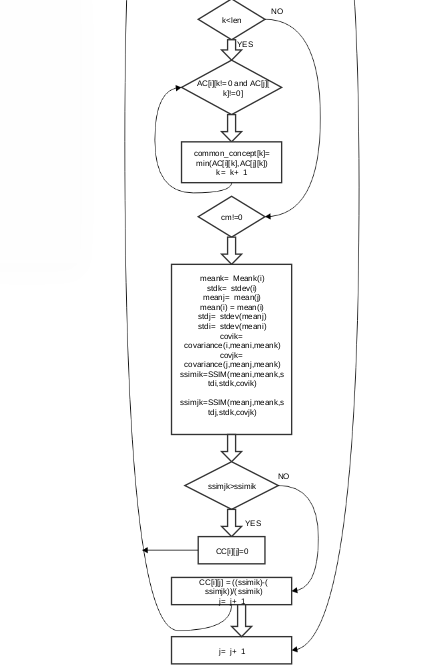


**2. Concept Extraction**



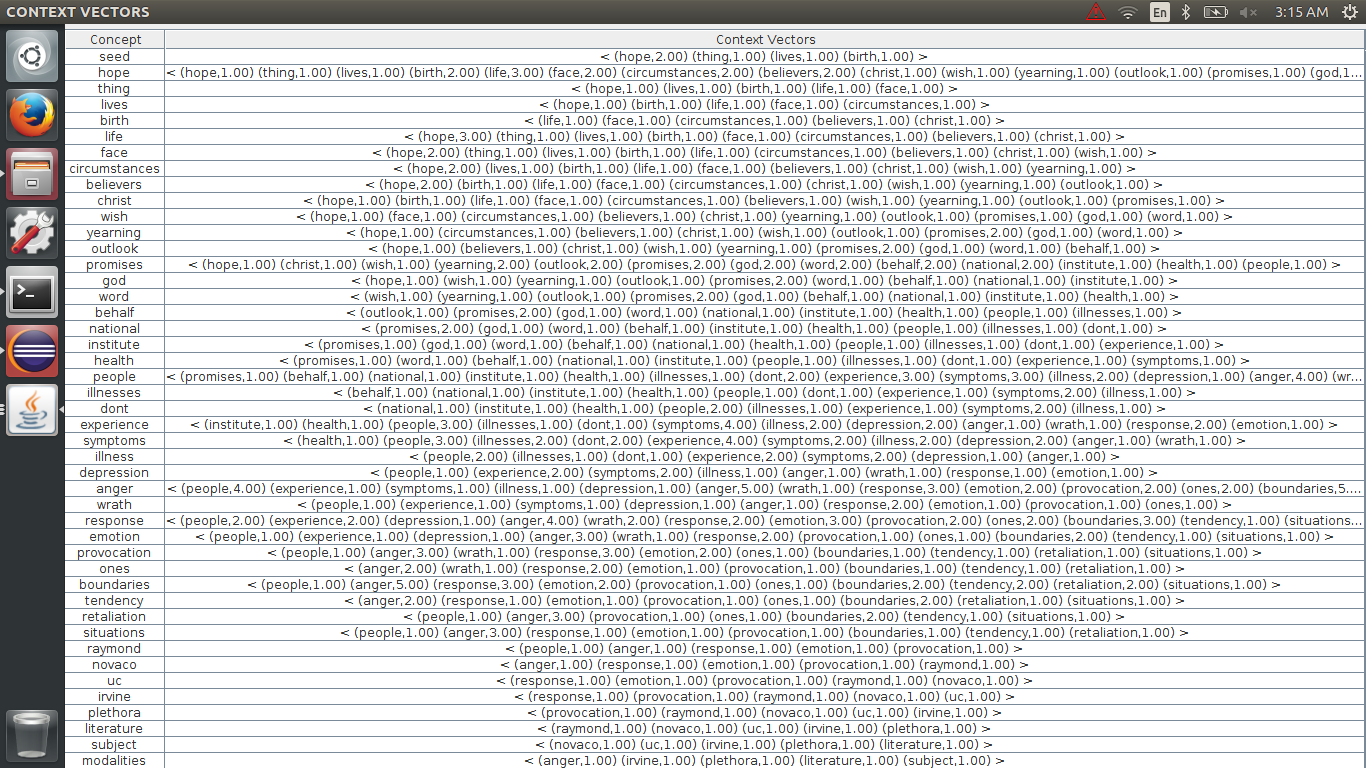
**3. FUZZY RELATION EXTRACTION:**





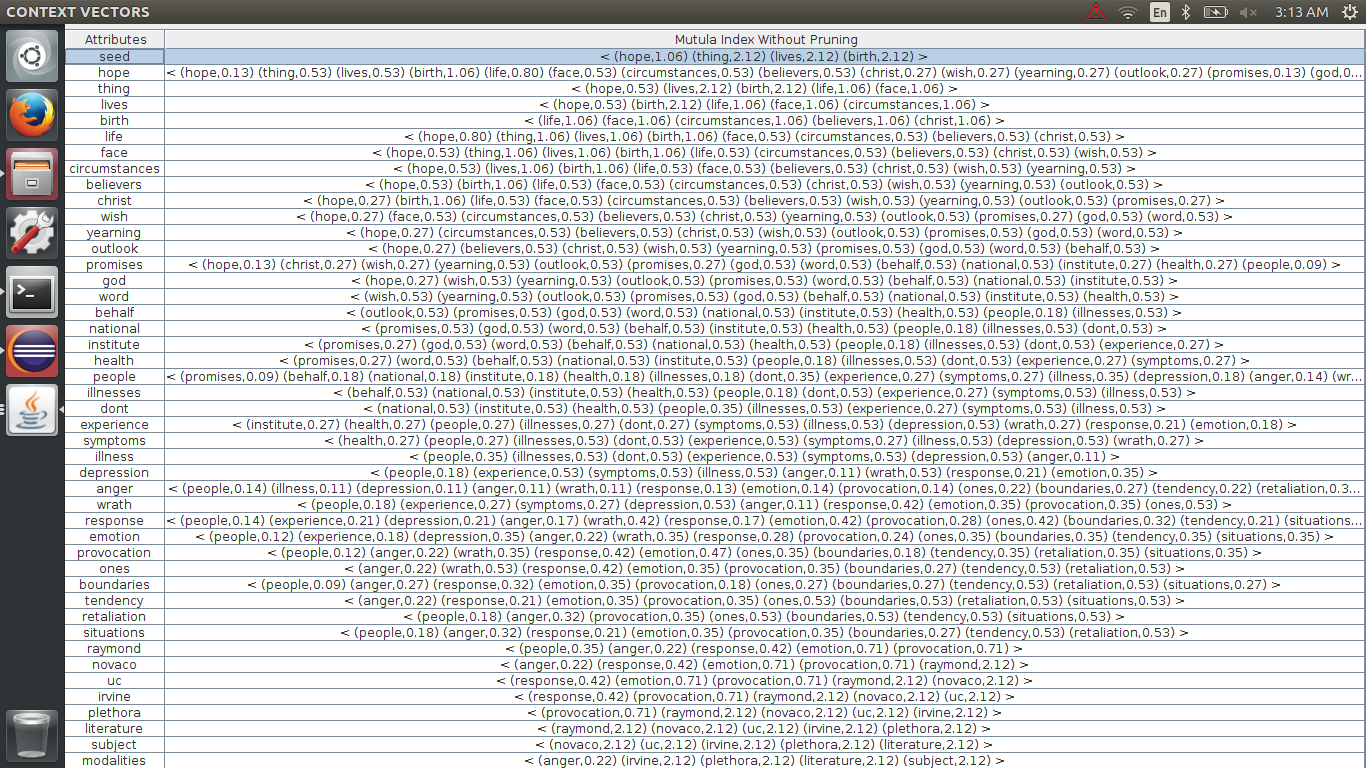
**RESULTS**

1. **Attribute-Attribute Matrix**



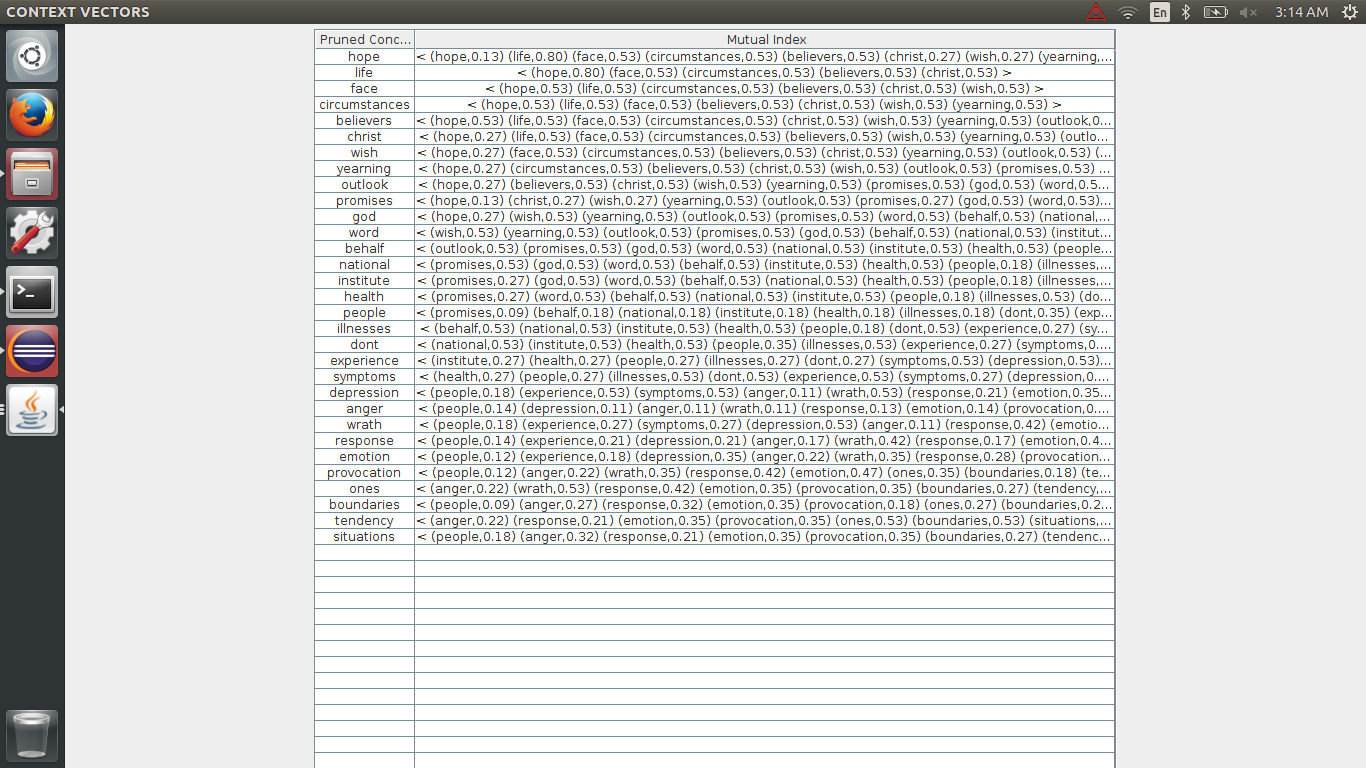
**Fig 17. Text Window Frequency Vector of Attributes.**

**2. Mutual Information Matrix (Probability matrix)**



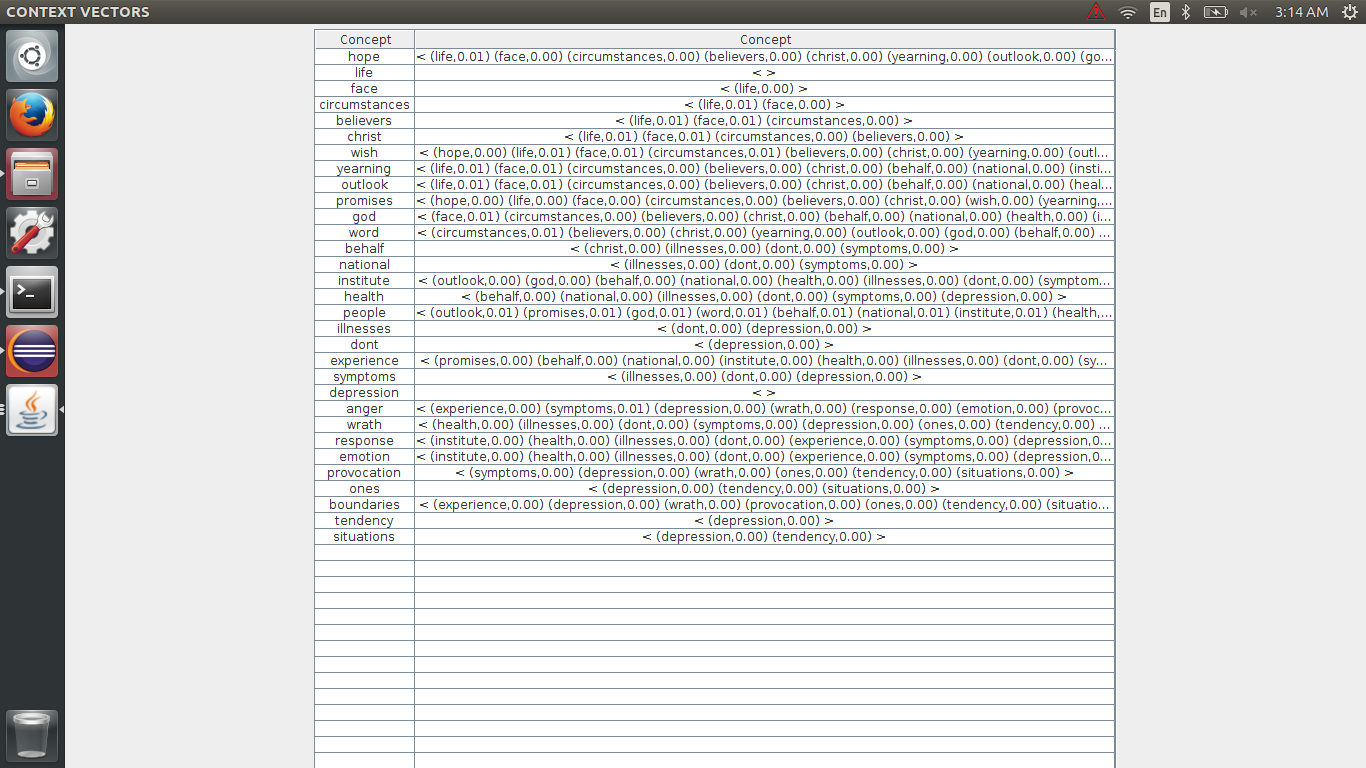
**Fig 18. Mutual Information among Attributes.**

**3. Pruned Concept From Mutual Information**



**Fig 19. Pruned Concepts**

**4. Concept- Concept Matrix**



**Fig 20. Mapping of Concepts and subconcepts.**

When the above mentioned algorithm was run on a small set of document, containing text on hope, peace and depression, the results were obtained in 4 matrixes, the final matrix maps the potential concepts with each others, some of which included experience, health, hope.

**CONCLUSION**

The algorithm tries to implement a concept map generation technique which is underpinned by

a context-sensitive text mining method and a fuzzy domain ontology extraction algorithm. The proposed mechanism can automatically construct concept maps.

Future work involves a larger scale of field test for the concept map generation mechanism. Other text mining methods can be integrated to apply this algorithm on variety of fields from political debates to bioinformatics to Wikipedia dumps.

**FUTURE WORK**

A major part of the project undertaken has been implemented, but there are some portions, the ones that help compare the performance of single system implementation over Hadoop system. The algorithm implemented so far has been tested on raw data sets, and implemented successfully on a single system. The future work includes:

1. On the part of Hadoop, we need to scale up the cluster to include more than 3 system (10 at least). A script can be developed to automate the repetitive and common steps for all nodes in the cluster.
2. On the part of algorithm, in order to reduce the dimensionality of the concept space, unsupervised mapping techniques such as SVD(singular value decomposition) can be applied.
3. Conversion of algorithm into MapReduce form. The Hadoop system runs job in MapReduce format. Thus, in turn to leverage the Hadoop’s computation we need to convert our code into its MapReduce equivalent.
4. Performance Comparision: Various empirical testing methods can be applied to reset the threshold values, and the results compared to obtain the best filters. Also, the performance of the algorithm in MapReduce form needs to be compared against the single instance algorithm. This step will highlight the need of Hadoop, and quantify the problem statement undertaken by the project.
5. Construction of Concept Graphs: Various open source platforms and java compatible libraries are available to automatically generate concept maps about a domain. These visually appealing maps provide even a novice with fair idea about the main themes in the corpus and their interrelation. Development of such Concept graphs upon the results obtained, will be the final step in completing the project.

**REFERENCES**

# **Toward a Fuzzy Domain Ontology Extraction Method for Adaptive e-Learning.**Published in: Knowledge and Data Engineering, IEEE Transactions on (Volume:21 ,Issue: 6 )

# **Probabilistic Topic Models for Learning Terminological Ontologies.** Published in:Knowledge and Data Engineering, IEEE Transactions on (Volume:22 , Issue: 7 )

# **The Role of Domain Knowledge in a Large Scale Data Mining Project.** Published in: Springer-Verlag Berlin Heidelberg 2002.

# **Defining architecture components of the Big Data Ecosystem.** Presented at IEEE conference on collaboration Technologies and Systems, May 2014.

1. **Mining Big Data: Current Status, and Forecast to the Future,** ACM SIGKDD Exploration Volume 14, Issue 2
2. **Towards Scalable Systems for Big data Analytics: A technology Tutorial.** Published in Access IEEE (Volume 2)
3. **Design Principle for Effective Knowledge Discovery from Big data,** presented at 6th European Conference on Software Architecture.
4. **Techniques, Applications and CHallenging Issues in Text Mining.** Published in International Journal of COmputer Science Issues, Vol 9, Issue 6.
5. **Topics and Terms Mining in Unstructured Data Stores,** presented at International Conference on Computational Science and Engineering.