TRIBHUVAN UNIVERSITY

INSTITUTE OF ENGINEERING

Pulchowk Campus

Thesis No.: 066/ MSI/ 612

Unsupervised Text Classification as Topic Annotation

By

Nitu Bharati

066/ MSI/ 612

Thesis

Submitted to

Masters of Science in Information and Communication Engineering,

Department of Electronics and Computer Engineering

November, 2011

Unsupervised Text Classification as Topic Annotation

By

Nitu Bharati

Thesis Supervisor

Prof. Dr. Shashidhar Ram Joshi

A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science in Information and Communication Engineering

Department of Electronics and Computer Engineering
Institute of Engineering, Pulchowk Campus
Tribhuvan University
Lalitpur, Nepal

November, 2011

COPYRIGHT ©

Recommendation

The undersigned certify that they have read and recommended to the Department of Electronics and Computer Engineering for acceptance, a thesis entitled **"Unsupervised Text Classification as Topic Annotation**", submitted by **Nitu Bharati** in partial fulfillment of the requirement for the award of the degree of "**Master of Science in Information and CommunicationEngineering**".

……

Supervisor: Prof. Dr. Shashidhar Ram Joshi

……

External Examiner: Suresh Kumar Regmi

Departmental Acceptance

The thesis entitled **"Unsupervised Text Classification as Topic Annotation",** submitted by **Nitu Bharati** in partial fulfillment of the requirement for the award of the degree of "**Master of Science inInformation and Communication Engineering**" has been accepted as a bonafide record of work independently carried out by her in the department.

Prof. Dr. Shashidhar Ram Joshi
Head of the Department
Department of Electronics and Computer Engineering,
Institute of Engineering,
Tribhuvan University,
Pulchowk, Nepal.

TRIBHUVAN UNIVERSITY

INSTITUTE OF ENGINEERING

Pulchowk Campus

Department of Electronics and Computer Engineering

Student's Declaration

I hereby declare that I am the only author of this complete work and that no sources other than the listed here have been used in this work.

_____

Nitu Bharati

Date: ……

Supervisor's Recommendation

I hereby recommend this dissertation prepared under my supervision by **Ms. Nitu Bharati** entitled "**Unsupervised Text Classification as Topic Annotation**" in partial fulfillment of the requirements for the degree of Master of Science in Information and Communication Engineering be processed for the evaluation.

_____

Prof. Dr. Shashidhar Ram Joshi

Date: ……

ACKNOWLEDGEMENT

I would like to give a deep gratitude to Prof. Dr. Shashidhar Ram Joshi, Head of Electronics and Computer Department, the supervisor of my thesis, for his precious suggestions and valuable support to pursue my research successfully.

My special thanks to Lecturer Sharad Ghimire, Coordinator of MSI program for the encouragement and co-operations throughout the research.

I express my sincere appreciation and graceful thanks to my friends who have directly and indirectly contributed in my research.

Nitu Bharati

066/ MSI/ 612

ABSTRACT

Multi-class Text Classification is the task of classifying a given text into one or more than one classes taken form a set of predefined classes. A class can be a topic of a text, for example, a class of any text about a movie can be ``entertainment''. In this research I investigate unsupervised learning to accurately identify the topic of a given text. The cost involved in labeling a large amount of data and availability of huge amount of unlabeled data makes unsupervised learning an ideal choice. The probabilistic algorithm used for text classification can be termed as topic modeling and is capable to extract multiple topics within a single text of a document. LDA model used in this report exploits co-occurrence patterns of words in documents to extract semantically meaningful probabilistic clusters of words called topics .Each of those clusters is labeled using the significant terms selected in each cluster. Semantic distance between the significant terms from the clusters and Wikipedia documents is measured to identify labels for each cluster.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS

TC	Text Classification
SVM	Support Vector Machine
LDA	Latent Dirichlet Allocation
LSA	Latent Semantic Analysis
PLSA	Probabilistic Latent Semantic Analysis
EM	Expectation-Minimization
MCMC	Markov Chain Monte Carlo

Chapter 1

INTRODUCTION

Text Classification (TC)is the task of classifying documents into one or more predefined categories. The categories can be the topic of the document. For example, we have a text "*Liverpool scored two goals in a football match*" and two category/classes "sports" and "politics". A TC system would classify above example into "sports". In more general form, given multiple categories $C = \{c_1, c_2, ..., c_n\}$, a classification system *S* and an uncategorized text *T,* the TC task is to assign for T one or more classes from *C*. The relation of *T* to the classes of *C* is determined by the content of *T*.

1.1 Scope

All our existing knowledge are being rapidly digitized and made available online, in the form of news, blogs, web pages, scientific articles, books and social network. As a result, to find the topic we are looking for is becoming difficult. One of the tools to extract relevant information from this huge online data collection is to use *search. Keywords* are fed into search engine and the documents related to the *keywords* are extracted. The keyword based search are powerful way of *information* extraction but lacks *thematic* retrieval of the relevant information. For example, in the above example a keyword based search might not retrieve "*Liverpool scored two goals in a football match*" or any other sports based articles if we search with keyword ``sport''. Rather than searching for the theme it searches for the presence of the keyword in the text. Thus, there is a certain need of tools to extract documents based on the thematic content which will certainly be a better way to explore and extract the available digitized content.TC is used to discover and annotate these large archives of documents with thematic information.TC is primarily used in hierarchical organization of web pages, keywords tagging of articles, organization of news articles according to its topics. It can also be used to filter text and also in word sense disambiguation [1].

1.2 Motivation

It is established that TC is a necessary tool to have. Our online experience on information retrieval would be highly effective and qualitative with TC. This is itself a huge motivation towards developing such system. The process involve in developing such system also drives high motivation. Not only the problem of solving TC is interesting but it is highly challenging and doing that in an unsupervised environment is even more exciting. With the supervised technique we would generate enough examples for each topic and then train a classifier with those examples. The trained classifier would then be able to classify unseen documents into different and preferably to their respective classes. Then what if we don't have annotated data? It is a difficult task to annotate data. To come around the problem of data annotation we have to employ an unsupervised method which can produce a model that classifies any text into respective classes. Along with creating such models many other problems need to be handled for example, same word can have multiple meanings. For example: Bank can be used as the bank where the money is deposited or it could be taken as the bank of the river. Thus, different machine learning and information retrieval technique has to be applied for achieving optimal result for the text classification problem. Apart from being challenging and interesting, the reason that the solution is not obvious made me highly motivated towards this topic.

1.3 Aims

The major aim of this research is to build a system that can correctly identify any given text as one of the predefined category. This research aims to achieve this goal in an efficient way, i.e. by using as minimal information as possible to represent any document.

Along with achieving multi-class text classification this research also aims to utilize non-labeled data to train the text classification system. A supervised setting needs plenty of labeled data to perform effectively, this research aims to achieve a comparable effectiveness by using less than half of such labeled data along with lots of unlabelled data.

1.3.1 Contributions

Following are the major contribution of the research:

1. The research successfully uses unlabelled data to train a classifier for text categorization. This is a major contribution as labeling of a data is a costly and time consuming task. For this a Latent Dirichlet Allocation (LDA) for the categorization of topics within the text is implemented.
2. Since the classification is done in an unsupervised manner, thus by default the model can be transferred easily to texts written in other languages achieving same standards of accuracy [2]. Although evaluation for other languages has not be carried out in this project but previous literature does show the transferability property of unsupervised algorithms across different language domain.
3. Along with the classification of text, the model also lists out the words that are important for each topic. These words can be used for task such as search engine optimization.
4. The research implements many effective filters to significantly reduce the dimensionality of the feature vector of input documents. The model still maintains high accuracy in classification even with the reduced dimensionality. The reduction in dimensionality is achieved by employing various filters like: word count, POS tag, and TF-IDF.
5. The research also uses concept of semantic distance to identify labels for each of the classified text. The implementation of this is done by calculating the cosine similarity between the top words of the topics and the Wikipedia documents.

Chapter 2

LITERATURE REVIEW

2.1 Related Work and Background

Methods used in TC can be classified into three categories. There categories are the settings under which the model is built, they are (1) supervised, (2) semi-supervised and (3) unsupervised. Following section describes the work done in each of the settings.

2.1.1 Supervised Classification:

In this setting the text classification model is learnt by training with the annotated data. Large volume of corpus is annotated with respective classes. The classification function *f* is learnt by generalizing over the pair of text and its respective classes. High accuracy can be achieved with this setting but on the downside the annotated process is rigorous, time consuming and costly. Accuracy is directly proportional to the amount of labeled data [3], thus without proper resources and time supervised setting becomes intractable. One of the most popular supervised algorithms is Support Vector Method (SVM) [4].

2.1.1.1 Support Vector Machine (SVM)

SVM is a supervised classifier which uses quadratic programming to classify the input data. Inherently SVMs are two-class classifiers. SVM uses labeled training data to build a classifier. A classifier is a hyperplane which separates two different classes of data. Mathematically, if we have a training data *D* with set of n points, defined as:

$$D = \{(\boldsymbol{x}_i, \boldsymbol{y}_i) \mid \boldsymbol{x}_i \in R^p, \ \boldsymbol{y}_i \in \{-1,1\}\}_{i=1}^{n} \qquad \text{equation (2.1)}$$

where, x_i represents the data points and y_i represents the class the data points belongs to. Since SVM is a binary classifier y_i can have two values -1 and +1.A hyperplane can be represented by the equation:

$$\boldsymbol{w}.\boldsymbol{x} - b = 0 \qquad \text{equation (2.2)}$$

Where, **w** is the normal vector from the hyperplane, and dot represents the dot product.

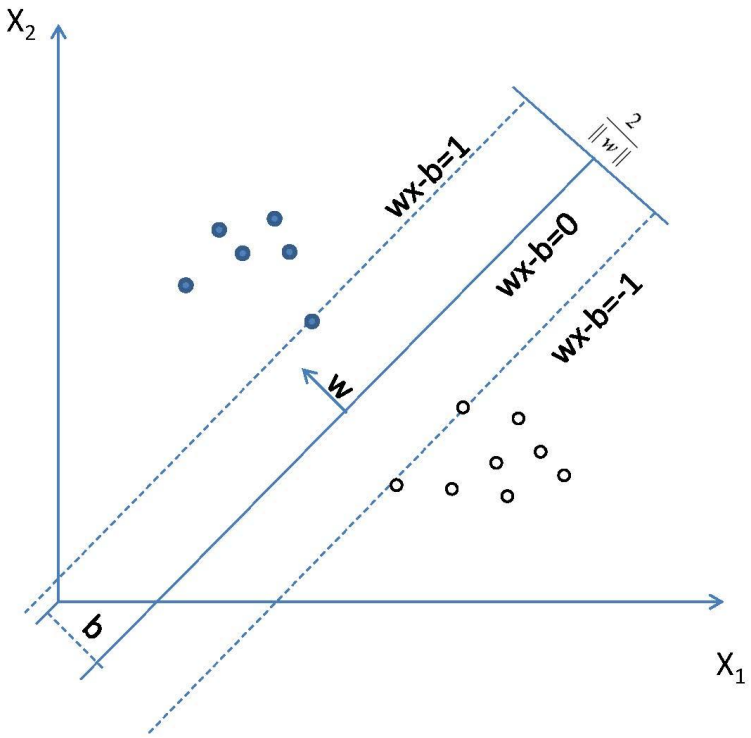

Fig2.1: SVM hyperplane

Any data points which satisfies the hyperplane equation $\boldsymbol{w}.\boldsymbol{x} - b = 1$ and $\boldsymbol{w}.\boldsymbol{x} - b = -1$ are called support vectors. These are the data points which are clearly separated into two classes. The optimization algorithm in the SVM forms a separating hyperplane of equation $\boldsymbol{w}.\boldsymbol{x} - b = 0$ by maintaining a maximum distance possible from the support hyperplane. In doing so, the newly created hyperplane separates the data into two different classes as shown in Fig2.1. Fig2.1 shows the data being represented in two-dimension, but in actual case SVM through its Kernel function represents data in multi-dimension and the separating data will also be multi-dimensional. Same logic can be used to train a multiclass SVM. Instead of classifying the data into -1 and +1 the hyperplane equation is represented as:

$$\boldsymbol{w}.\boldsymbol{x} - b = 1 - ב \qquad \text{equation (2.3)}$$

Where,ב represents the margin between the value of the correct class and the nearest other class. After creating the hyperplane from the training data, it is used in unlabelled data to classify them. Thus during classifying the actual class will be the class which gives the maximum value for the equation

$\boldsymbol{w}.\boldsymbol{x}$i.e.

$$y = \max(\boldsymbol{w}.\boldsymbol{x}) \qquad \text{equation (2.4)}$$

2.1.2 Semi-supervised

A document classification can be considered as a function which assigns a value Y to an input X, this can be represented as:

$$f : X \to Y \qquad \text{equation (2.5)}$$

In a supervised setting such a function is learned from a data pair, where each pair consists of an input x and its output y.

(x_i, y_i) where i runs from 1 to n, n=total number of training data.

In an unsupervised setting such a function is learned from a data input consisting of just input values, i.e. the x's. A clustering algorithm is used to group these data into different clusters where each clusters represents a category.

It has been always shown that the accuracy of the supervised algorithms is higher than the accuracy of unsupervised algorithm. Thus using labeled data is a good thing. But on the other hand preparing such labeled data can be time consuming and can take lots of effort. Semi-supervised setting uses both labeled data and unlabelled data. In semi-supervised setting above function is learned by using both labeled data pair and unlabeled data. The input to any semi-supervised function will be:

(x_i, y_i) where i runs from 1 to n, n=total number of training data and

(x_i) which is the unlabelled data.

This can be represented more clearly from the table below:

Table 2.1: Document Feature Representation

Y	X_1	X_2	X_3	X_4
-1	0	0	1	1
+1	0	1	0	0
-1	0	0	1	0
?	1	1	0	0
?	0	1	1	1

Table 2.1 represents a sample training data for a document classification system. Each row represents a document, where Y is the class/category of the document and X is the feature representation of the document. For example, a document with feature {0, 0, 1, 1} is of category -1. It can be seen that the first three documents class is defined, but the last two does not have its class defined. The last two are the unlabeled data and in semi-supervised setting we use the unlabeled data in conjunction with the labeled data to train the classifier.

The simplest way to achieve the semi-supervised learning from above data completes in two steps:

Train a classifier from the labeled data only (first three rows in above example)

Use the trained classifier to label the unlabeled data (last two rows) and use the whole set to train the classifier again.

The result of step one will be a classifier C_1, which we will use to label the last two rows, thus the above table after using C_1 to label last two rows might look like as shown in Table 2. Now the whole feature dataset is used i.e. all the five rows to train a new classifier. In this way the unlabeled data is used to train a document classification system.

It can be seen that the approach is simple and logical, but there is still one major flaw in the approach, how can we be sure that the label which are obtained for the unlabeled data from

the first step are the correct ones? This problem can be rectified by using a popular semi-supervised approach called co-training. But semi-supervised setting also suffers with the same flaw as supervised setting, i.e. to attain a high accuracy the initial seed labeled data has to be quite high. This is considerably less than supervised setting but still for decent accuracy number of labeled data has to be high.

Table 2.2: Document feature representation after labeling the unlabeled data

Y	X_1	X_2	X_3	X_4
- 1	0	0	1	1
+ 1	0	1	0	0
- 1	0	0	1	0
+ 1	1	1	0	0
- 1	0	1	1	1

2.1.3 Unsupervised setting:

In an unsupervised learning scenario there is no labeled data and thus no direct feedback on the actions of a classifier. Instead the classifier tries to make a good representation of the input vector in the output and a task-independent measure is used to determine the quality of such representation. For example in Web clustering [5] if we have a large number of documents, but we do not strictly know the full classification structure we want to classify them into, unsupervised learning can be used. What unsupervised learning would do is simply grouping together web pages with similar content or topic. The quality measure could be the confidence of classification given a fixed number of categories to use.

There are many popular algorithms that can be used for the text categorization problem. The popular ones are LDA, LSA, and other probabilistic algorithms [6] [7].Latent topic modeling has become very popular as a completely unsupervised technique for topic discovery in large document collections. These models, such as PLSA [8] and LDA [9] exploit co-occurrence patterns of words in documents to extract semantically meaningful probabilistic clusters of

words called topics. These models also assign a probabilistic membership to documents in the latent topic-space, allowing us to view and process the documents in this lower-dimensional space. These probabilistic models are both generative process and follow a natural way of document construction. Due to this fact these models become default selection for topic modeling. In my thesis, I prefer LDA over PLSA because with the addition of Dirichlet prior on the LDA model the overall mathematical calculation is hugely lesser than PLSA along with increase in accuracy.

2.1.4 Feature Selection

In a typical text classification approach for the classification of text articles in English, the word frequency is used as the primary part of the feature vector. This typically produces feature vectors with dimensionality in the order of tens of thousands of dimensions [10]. The computational complexity of any operations with such feature vectors will be proportional to the size of the feature vector [11], so any methods that reduce the size of the feature vector while not significantly impacting the classification performance are very welcome inany practical application. Additionally, it has been shown that some specific words in specific languages only add noise to the data and removing them from the feature vector actually improves classification performance [11].

The set [12,10] of feature reduction operations involves a combination of three general approaches:

1. Stop words;

2. Stemming;

3. Statistical filtering.

In any language there are many words that convey little or no meaning, but are required by the grammar structure of the language; these words are called “stop words”. As an example in the English language words like: “a”, “the”, “but” and many others are considered to be such stop words. It is common practice to exclude stop words from the feature vector. Stop word lists can be used or the stop word can be determined from their frequency, which is said to be more efficient and language independent [13,14].

Second way of traditional feature reduction is the use of stemming to reduce frequencies of words with a common root to a single feature, for example, if the document would contain 7 instances of the word “house”, 3 instances of the word “houses” and 2 instances of the word “housing”, then after stemming reduction these three separate features would be reduced to only one that would describe that words with the root similar to “house” occurred in the document 12 times (7+3+2). Traditionally Porter's algorithm [15] is being used for stemming in English.

Statistical filtering practices are used to select those words that have higher statistical significance. Many different statistical methods are being researched and used for feature vector filtering, but the main difference between these methods is how much information about the source data is being used. It is possible to calculate generic statistical significance of a word in relation to how different its use frequency is in different documents, but more sophisticated algorithms also take into account the proposed classification of said documents and are essentially computing statistical significance of words in specific categories.

Most represented [10] statistical filtering approaches are: odds ratio, mutual information, cross entropy, information gain, weight of evidence, $\chi 2$ test, correlation coefficient [17], conditional mutual information maximin[18], and conformity/uniformity criteria [19]. Yang [11]compared some of those methods. In simple terms, most formulas give high scores to words that appear frequently within a category and less frequently outside of a category (conformity) or to the opposite (non-conformity). And additionally higher scores are given to words that appear in most documents of a particular category (uniformity).

Another way of reducing the feature vector is through the use of genetic computing [20]. However, in most applications the use of genetic computing will use up more resources then the resulting feature reduction could spare during the production run of the system.

After a good application of feature reduction algorithms one can expect to bring the size of a typical feature vector down from hundreds of thousands of dimensions to a few thousands of dimensions. However some research [21] shows that during feature reduction some subtle information is lost that could be useful for enhancing the precision of classification. So feature reduction in most cases is a speed versus precision trade off.

2.2 Conclusions

Text classification is an interesting field of research that has been revitalised by the increase of information flow available. It has seen large attention especially due to the high growth rate of Internet and the importance of Internet search engines and generic classification of content on the Web. Process of text classification is well researched, but still many improvements can be made both to the feature preparation and to the classification engine itself to optimise the classification performance for a specific application. The different kinds of algorithms for text classification are discussed. The primary problem with the supervised and semi-supervised algorithms was that the number of labelled data required is quite high. Thus for every domain adaptation the cost will be quite high. The unsupervised setting is described and it is concluded that the research down this line is a better path to select for solving text classification problem.

Chapter 3

RESEARCH METHODOLOGY

3.1 Introduction

A topic model is a generative model for documents. Such model explains the process of creating documents. With topic models new documents are created using simplified probabilistic methods. The probabilistic method starts by choosing a distribution over topics. Then according to this topic distribution, a topic is sampled randomly and then from the word-topic distribution words are drawn for that topic. The collection of such words forms a document. The interesting fact about topic models is that standard statistical techniques can be applied to invert the document creation process, i.e. by inverting the generative process we can infer the set of topics that were responsible for generating a collection of documents.

For example: we have 3 topics namely, topic1, topic2 and topic3 as shown in Figure 1. The figure shows the ten words that have the highest probability under each topic. The words in these topics are related to sports, science and technology, and health. By choosing different distributions over topics, documents with different content can be generated. For example, by giving equal probability to the first two topics, one could construct a document about a person who plays cricket match in computer and by giving equal probability to last two topics, one could construct a document about a person suffering from headache due to longer use of laptop for the work.

Topic 1

Word	prob.
PLAY	0.067
MATCH	0.060
CRICKET	0.054
FOOTBALL	0.050
RUN	0.045
RUNS	0.043
GOAL	0.032
FOUR	0.023

Topic 2

Word	prob.
TECHNOLOGY	0.043
APPS	0.032
WINDOWS	0.024
MICROSOFT	0.021
APPLE	0.017
GOOGLE	0.012
DEVICE	0.010
WIRELESS	0.008

Topic 3

Word	prob.
MEDICINE	0.053
SURGERY	0.046
BLOOD	0.034
HEARTS	0.031
DOCTOR	0.028
DOCTORS	0.026
HOSPITAL	0.016
PATIENT	0.012

Fig 3.1: Illustration of three hypothetical topics

3.2 Generative Model

Each topic is individually interpretable, providing a probability distribution over words that pick out a coherent cluster of correlated terms [7].A generative model for documents is based on simple probabilistic sampling rules that describe how words in documents might be generated on the basis of latent (random) variables [9]. While implementing generative model, the target is to determine the set of latent variables that can best describe the observed words of the document.

The generative process is illustrated in Figure with two topics. Topics 1 and 2 are thematically related to sports and technology and are illustrated as bags containing different distributions over words. Different documents can be produced by picking words from a topic depending on the weight given to the topic. For example, documents 1 and 3 were generated by sampling only from topic 1 and 2 respectively while document 2 was generated by an equal mixture of the two topics. The superscript numbers associated with the words in

documents indicate which topic was used to sample the word. The generative process described here does not make any assumptions about the order of words as they appear in documents. The only information relevant to the model is the number of times words are produced. This is known as the *bag-of-words assumption*.

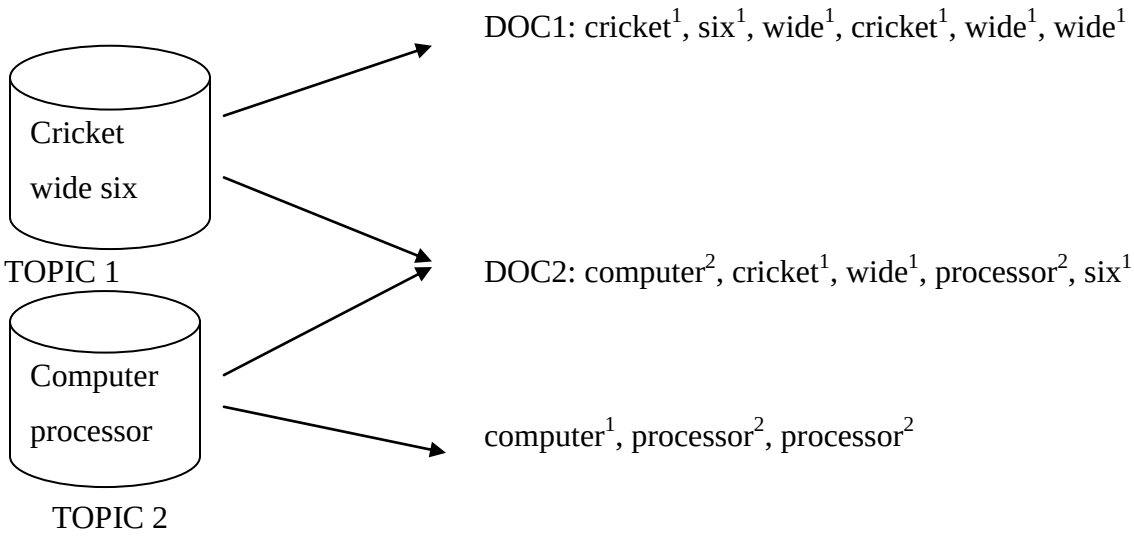

Fig 3.2: Illustration of the generative process underlying the topic model

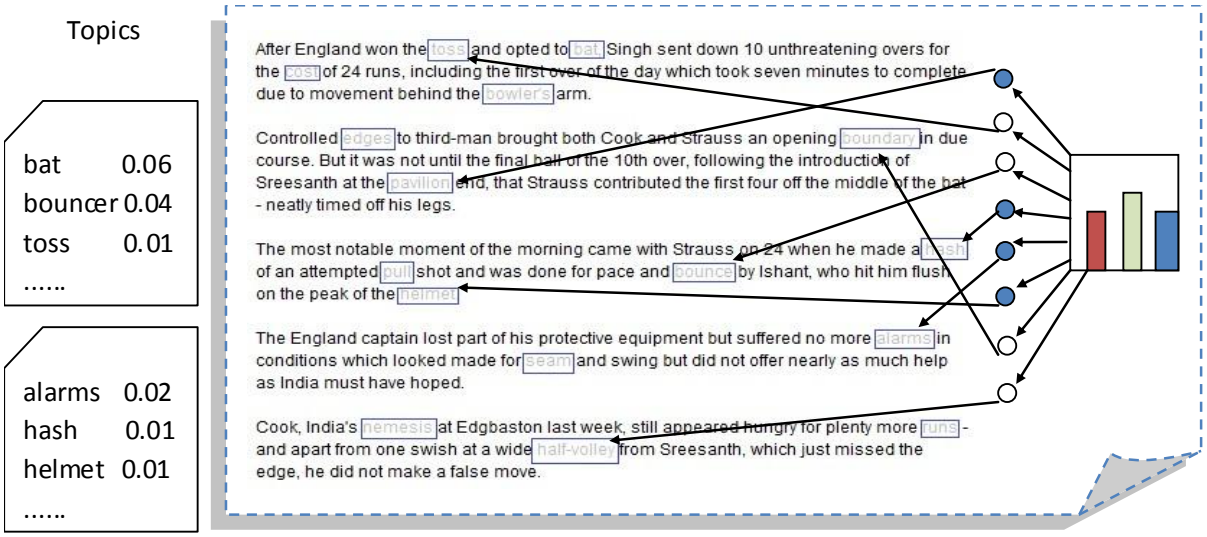

Fig 3.3:An example of Generative Process

For the generation of each document a distribution is assumed over topics (the histogram at right in Figure 4). From this distribution topics are chosen (the colored coins) and then the word from the corresponding topic is chosen. Words are chosen from the distribution over words for each topic (boxes at the leftmost part). The topics and the topic assignment in this

figure are ``sports'' and ``products''. The distributions shown in the figure are hypothetical and are not derived from the real data. Real distributions and values will be shown later in the report.

It can be seen in Fig the example process of a generative model. A *topic* is formally defined to be a distribution over a fixed vocabulary. For example the *sports* topic has words about sports with high probability and the *products* topic has words about products with high probability. It is assumed that the topics are generated first, before the documents. Now for each document in the collection, the words are generated in a two-stage process.

1. Randomly choose a distribution over topics.
2. For each word in the document
    a. Randomly choose a topic from the distribution over topics in step 1.
    b. Randomly choose a word from the corresponding distribution over the vocabulary.

This statistical model reflects the intuition that documents exhibit multiple topics. Each documents exhibits the topics with different proportion (step 1); each word in each document is drawn from one of the topics (step 2), where the selected topic is chosen from the per-document distribution over topics (step 2a).

3.2.1 Inference

The above algorithmic explanation describes how a document is generated. Now, as it have been pointed in the introduction section the main aim of the topic modeling is to discover the topics that are within the documents. The documents itself are observed, the latent variables are the *topics* that resides within the document i.e. per-document topics distribution and the *topic word association* i.e. per-document per-word topic assignment. Thus the computation task of the topic modeling is to *use the observed documents* to *infer the hidden topic structure.* This can be thought of as reversing the generative process i.e. what are the *hidden variables* that likely generated the observed collection?

3.2.2 Creating model

The unsupervised method of topic modeling is to observe huge collection of documents and apply probabilistic implementation to *infer* topics within all those documents and its words. The mathematical implementation of inference operation will be discussed in detail in later sections. The algorithm works such that it becomes more certain of the topics that it infers as it sees more documents, so with every *next* document its inferring ability becomes better.

Once topics are inferred for all the documents iteratively and a model is obtained that can infer topics with high accuracy. This phase is the training phase.

Since the observed collection of data is huge, the model thus created is general enough to infer topics for now unseen documents (which actually for the algorithm is just the *next* document).

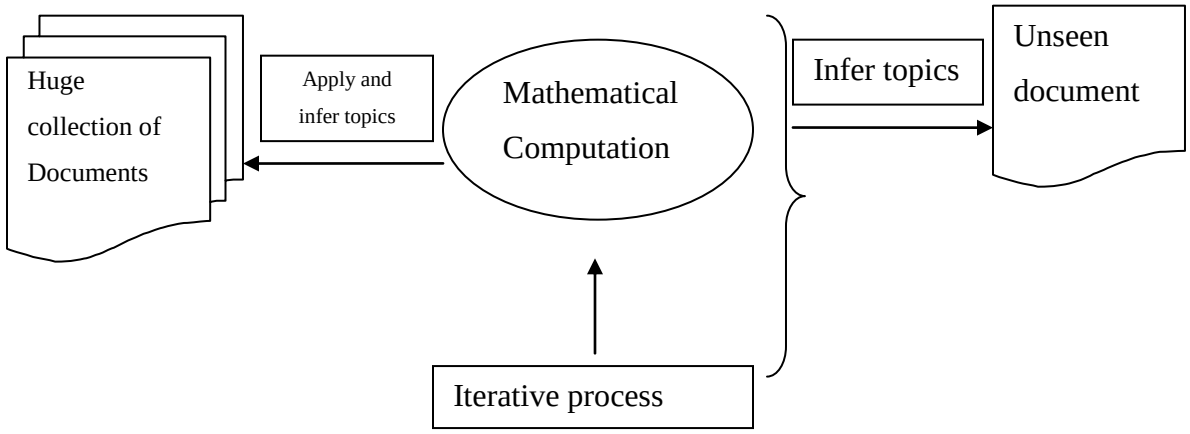

Fig 3.4 : Training and application phase of generative model

The above figure is the pictorial representation of the training phase and the application phase.

Next the mathematical representation of the topic modeling will be discussed, Latent Dirichlet Allocation (LDA). LDA is the simplest and effective topic model. In LDA the distribution that is used to draw the per-document topic distributions in step 1 (the graph in Fig) is called a *Dirichlet* distribution. In the generative process for LDA, the result of the Dirichlet is used to *allocate* the words of the document to different topics. The topics in the documents are *latent* (hidden) and the algorithm tries to estimates the hidden variable.

3.3 LDA and Probabilistic Topic Model

LDA and other topic models are part of the larger field of *probabilistic modeling*. In generative probabilistic modelling, we treat our data as arising from a generative process that includes hidden variables. This generative process defines a *joint probability distribution* over both the observed and hidden random variables. We perform data analysis by using that joint distribution to compute the *conditional distribution*of the hidden variables given the observed variables. This conditional distribution is also called the *posterior distribution*.

LDA falls precisely in this framework, with the addition of Dirichlet prior on the topic distribution. Mathematically,

A document is a mixture of topics. Suppose P(t) be the distribution over topics t in a document and P(w | t) for the probability distribution over words w given topic t. Several topic-word distributions P(w | t) were illustrated in Figures 1 and 2, each giving different weight to thematically related words. Each word w_i in a document (where the index refers to the *i*th word token) is generated by first sampling a topic from the topic distribution, then choosing a word from the topic-word distribution. $P(t_i = j)$ is taken as the probability that the *j*th topic was sampled for the *i*th word token and $P(w_i | t_i = j)$ as the probability of word w_i under topic *j*. The model specifies the following distribution over words within a document:

$$P(w_i) = \sum_{j=1}^{T} P(w_i | t_i = j)\, P(t_i = j) \qquad \text{equation (3.1)}$$

Where, *T* is the number of topics. Let $\varphi^{(j)} = P(w \mid t = j)$ refer to the multinomial distribution over words for topic *j* and $\theta^{(d)} = P(t)$ refer to the multinomial distribution over topics for document *d*. Now, let us assume we have *D* text documents and each document *d* consists of N_d word tokens. Let *N* be the total number of word tokens (i.e., $N = \Sigma N_d$). The parameters φ and θ indicate which words are important for which topic and which topics are important for a particular document, respectively.

Now for LDA we place a Dirichlet prior on θ [6]. As a conjugate prior for the multinomial, the Dirichlet distribution is a convenient choice as prior, simplifying the problem of statistical inference. The Dirichlet prior α for a topic can be interpreted as prior observation count for number of times that topic is sampled in the document, before having observed any

actual words from that document. A symmetric Dirichlet (β) prior is also placed on φ that can be interpreted as the prior count on the number of times words are sampled from a topic before any words from the corpus is observed.

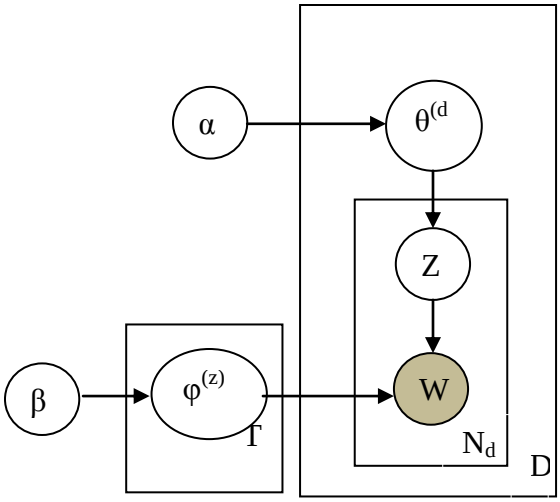

Fig 3.5 : Plate Diagram of Generative Model

The above discussed mathematical model can be represented as in plate notation [22] as shown in Figure 6. In plate notation the shaded variables represent the observed variable and unshaded variables represent the latent variables. The plate (rectangular box) represents the repetition of sampling steps with number of samples at its lower right corner. The arrows represent the conditional dependencies between the variables. The plate surrounding $\varphi^{(z)}$ illustrates the repeated sampling of word distributions for each topic *z* until *T* topics have been generated. The inner plate over *z* and *w* illustrates the repeated sampling of topics and words until N_d words have been generated for document *d*. The plate surrounding $\theta^{(d)}$ illustrates the sampling of a distribution over topics for each document *d* for a total of *D* documents.

3.4 Gibbs Sampling

Hofmann (1999) used the expectation-maximization (EM) algorithm to get the direct estimates of the topic-word distributions, φ and the topic distributions, θ for each document. But this approach had to face the problems involving local maxima of the likelihood function. Thus, rather than direct estimation of φ and θ, new approach was devised that

involves direct estimate of the posterior distribution over z (the assignment of word tokens to topics), given the observed words w, while marginalizing out φ and θ. Each z_i gives an integer value [1..T] for the topic that word token i is assigned to. Because many text collections contain millions of word token, the estimation of the posterior over z requires efficient estimation procedures. For that an algorithm is used that implements Gibbs Sampling that is a form of Markov chain Monte Carlo (MCMC), which is easy to implement and provides a relatively efficient method of extracting a set of topics from a large corpus. Markov Chain Monte Carlo (MCMC) refers to a set of approximate iterative techniques designed to sample values from complex (often high-dimensional) distributions [22].

Gibbs Sampling is a specific form of MCMC that involves simulating a high-dimensional distribution by sampling on lower-dimensional subsets of variables where each subset is conditioned on the value of all others. The sampling is done sequentially and proceeds until the sampled values approximate the target distribution. Direct estimates of φ and θ is not provided implementing Gibbs Sampling, rather φ and θ are approximated using posterior estimates of $\boldsymbol{z}$

The collection of documents is represented by a set of word indices w_i and document indices by d_i, for each word token i. The Gibbs sampling procedure considers each word token in the text collection and estimates the probability of assigning the current word token to each topic, conditioned on the topic assignments to all other word tokens. From this conditional distribution, a topic is sampled and stored as the new topic assignment for this word token. Suppose the conditional distribution is represented as $P(\ z_i{=}j \mid \boldsymbol{z}_{-i},\ w_i,\ d_i,\ \cdot\ )$, where $z_i = j$ represents the topic assignment of token i to topic j, $\boldsymbol{z}_{-i}$ refers to the topic assignments of all other word tokens, and "·" refers to all other known or observed information such as all other word indices $\boldsymbol{w}_{-i}$ and document indices $\boldsymbol{d}_{-i}$, and hyper parameters α, and β.

Griffiths and Steyvers[7] showed how this can be calculated by:

$$P(z_i = j|z_{-i}, w_i, d_i, .) \propto \frac{C_{w_i j}^{WT} + \beta}{\sum_{w=1}^{W} C_{wj}^{WT} + W\beta} \frac{C_{d_i j}^{DT} + \alpha}{\sum_{i=1}^{T} C_{d_i t}^{DT} + T\alpha} \qquad \text{equation (3.2)}$$

where $\mathbf{C}^{WT}$ and $\mathbf{C}^{DT}$ are matrices of counts with dimensions W x T and D x T respectively. C_{wj}^{WT} contains the number of times word w is assigned to topic j, not including the current instance i and C_{dj}^{DT} contains the number of times topic j is assigned to some word token in document d, not including the current instance i. Above equation gives the probability that is not in normalized form. Thus to get actual probability of assigning a word token to topic j, the quantity in Equation 3 for topic t is divided by the sum over all topics T.

The topic assignments for a particular word token are affected by various factors which can be understood by examining the two parts of above equation. The left part is the probability of word w under topic j whereas the right part is the probability that topic j has under the current topic distribution for document d. If many word tokens are assigned to topic j (across documents), it will increase the probability of assigning that particular word token to topic j. At the same time, if topic j has been used multiple times in one document, it will increase the probability that any word from that document to be assigned to topic j. Therefore, words are assigned to topics depending on two factors- how likely the word is for a topic and how dominant a topic is in a document.

The Gibbs sampling algorithm starts by assigning each word token to a random topic in [1..T]. For each word token, the count matrices $\mathbf{C}^{WT}$ and $\mathbf{C}^{DT}$ are first decremented by one for the entries that correspond to the current topic assignment. Then, a new topic is sampled from the distribution in Equation 3 and the count matrices $\mathbf{C}^{WT}$ and $\mathbf{C}^{DT}$ are incremented with the new topic assignment. Each Gibbs sample consists the set of topic assignments to all N word tokens in the corpus, achieved by a single pass through all documents. During the initial stage of the sampling process (also known as the burnin period), the Gibbs samples have to be discarded because they are poor estimates of the posterior. After the burnin period, the successive Gibbs samples start to approximate the target distribution (i.e., the posterior distribution over topic assignments). At this point, to get a representative set of samples from this distribution, a number of Gibbs samples are saved at regularly spaced intervals, to prevent correlations between samples.

The sampling algorithm gives direct estimates of z for every word. However, as per the necessity φ' of the word-topic distributions can be estimated by using count matrix as follows:

$$\varphi_i'^{(j)} = \frac{C_{ij}^{WT} + \beta}{\sum_{k=1}^{W} C_{kj}^{WT} + W\beta} \qquad \text{equation (3.3)}$$

3.5 Data Set

All the data is collected manually from Google news site[1]. From the news index only news from following category were extracted, namely Entertainment, Politics, Sports and Business. Total of 900 news article were collected. The training set and test set for each category are both sampled from these 900 news article. It can be seen that the document size is considerably large; this is a desirable feature because a small size document will lead to make feature space sparse and LDAs are less effective in very sparse feature space.

3.6 Preprocessing

All the words in the document are converted into its root grammatical form. To get the root form of each word TreeTagger[2] is used. TreeTagger is a tool for annotating text with part-of-speech and lemma information. It was developed by Helmut Schmid in the TC project at the Institute for Computational Linguistics of the University of Stuttgart. Converting the words into lemma forces the words with same meaning to act as a single feature. For example; word ``played'' and ``play'' should be represented as same feature, since my intuition is that the forms of the words does not play a major role in disambiguating the category of a document. Also all the words in the document are converted to lowercase.

3.7 Filters

A bag of words representation is implemented as the feature vector. In this representation each word in the document is considered to uniquely identify the category of the document. For example, a document with words such as ``basketball'', ``football'' etc. will be of

[1] http://news.google.com/

[2] http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/DecisionTreeTagger.html

category *Sports* and a document with words such as ``share'', ``market'' etc. will be of category *Business.* This is very logical but not all the words in the document represent the category of the document. For example, words like ``is'', ``what'', ``being'' etc. can be present in any category. If such words are included in our feature vector, then it can confuse the model and can lead to errors, since the feature vector for different categories will be less distinct with presence of these common words. Also removing these words makes the document size smaller, thus decreasing training time. Thus, three different kinds of filters are implemented to remove unwanted words from documents. The filters are word count, POS tag and TF-IDF. Each filter is explained in following section.

3.7.1 Word Count

Words which occur very infrequently throughout a category, plays insignificant role in uniquely identifying a category. Thus, such words can be safely removed from the documents. One thing to be careful is that the frequency of the words is not counted in each document but is counted throughout all the documents of each category. For example the word ``weekend'' occurred 4 times throughout the 250 documents of Business category, thus this word can be removed from the documents of business category. The threshold for the word is set to 5 (decided through careful observation) i.e. all words in each category with frequency less than 5 is removed.

3.7.2 TF-IDF

TF-IDF (Term Frequency-Inverse Term Frequency) is a statistical measure to evaluate how important a word is to a category in a collection of document of same category. The importance of the word in a category increases as the frequency of the word in the category increases but the importance is inversely proportional to the number of documents the word occurs in.

For example, the word ``is'', in category Sports. The word will occur many times throughout all documents, thus its *term frequency* is very high, and also the word will occur in almost all the documents, thus its *document frequency* is also very high. TF-IDF will punish such words. On the other hand in the same category Sports the word ``football'' can have relatively higher term frequency but the word might not occur in all the document thus will

have low document frequency. TF-IDF will consider such words as being important to the category. Hence an *inverse document frequency* factor is incorporated which diminishes the weight of terms that occur very frequently in the collection and increases the weight of terms that occur rarely. Mathematically, term frequency is defined as:

$$tf_{i,j} = \frac{n_{i,j}}{\sum_k n_{k,j}} \qquad \text{equation (3.4)}$$

where, $n_{i,j}$ is the number of occurrences of the considered term (t_i) in document d_j, and the denominator is the sum of number of occurrences of all terms in document d_j. The denominator is used to normalize the total count of the words to prevent a bias towards longer documents (which may have a higher term count regardless of the actual importance of that term in the document).

And inverse term frequency is defined as:

$$idf_i = log \frac{|D|}{|\{d: t_i \in d\}|} \qquad \text{equation (3.5)}$$

where,

$|D|$ is the total number of document in the category (in our case 100)

$|\{d: t_i \in d\}|$ is the number of documents where the term t_i appears

Thus, finally,

$$(tf - idf)_i = tf_{i,j} \times idf_i \qquad \text{equation (3.6)}$$

A high weight in tf–idf is reached by a high term frequency (in the given document) and a low document frequency of the term in the whole collection of documents; the weights hence tend to filter out common terms.

For each category the tf-idf value of each word is calculated, and sorted the word tf-idf list in descending order, and then the words from the document which matched the top 2% word in the sorted list are removed

Table 3.1: POS tags irrelevant to document classification

POStag	Examples	POStag	Examples
AUX	do,done,have,is	WDT	to,thatwhat,which
CC	and,both,either	PDT	all,both,half
CD	0.5,1	POS	”s
DT	all,an,the	PRP	hers,herself,him
EX	there	PRP$	her,his,mine
FW	jeux	RP	along,across
IN	astride,among,whether	SYM	&

3.7.3 POS tags

Not all the words can uniquely identify the document. There are some words with certain part of speech which plays no significance for disambiguating a document category. Words with such part of speech can be removed from the document. To remove such words, first each document is annotated with the part of speech of every word. The POS tagging is done by using Stanford's POS tagger[3]. The tag symbols all have their standard meaning and are taken from Penn Treebank Tagset[4]. Then all the words from the document which has POS tags also presentment in predefined filter POS tag list is removed. Table show the entire POS tag list with example which are irrelevant for text classification. The POS tag list is acquired by careful evaluation of data set.

3.8 Feature Vector Representation

The project uses a bag of words representation i.e. each word in the document is a feature for that document. The feature is used as presence instead of count i.e. for all the feature is test is done to check if the feature exits in the document or not, its count in document is disregarded. Now different feature types used are discussed below.

3.8.1 Unigram

Unigram is the single word. Each unique single word is extracted from the whole category.

[3] http://nlp.stanford.edu/software/tagger.shtml

[4] http://www.ims.uni-stuttgart.de/projekte/CorpusWorkbench/CQP-HTMLDemo/PennTreebankTS.html

For now only unigram is implemented in the system. Using of context capturing features like bigram and sub-sequences are not used because the disambiguation of the words meaning for different context occurs during the inferring process.

3.9 Example Implementation

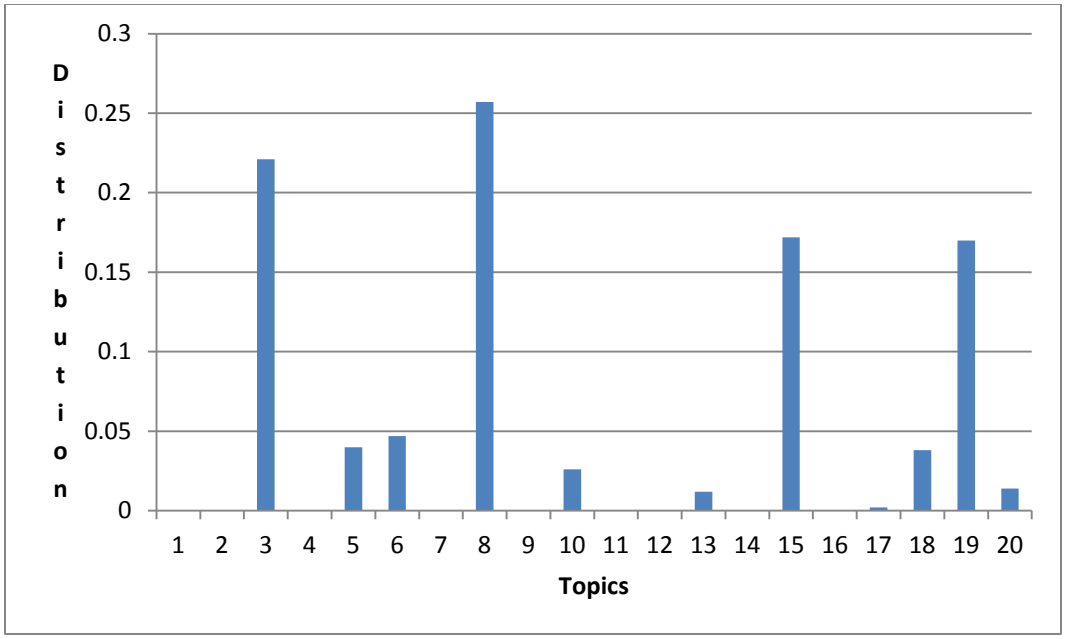

Fig 3.6: Distribution of topic in a document

The dataset as collected from the process described above is passed through all the filters, also described in above sections. The cleaned dataset is converted into the format suitable for the LDA implementation used. The resulting corpus contains one document per line, where each line includes 3 sections, as, (1) document ID (unique and randomly generated), (2) document label (irrelevant in my case, thus marked "X") and (3) the document content itself.

The Gibbs sampling process was run with the model containing the corpus, α=0.01, β=0.01, iteration=1000 and topics=20. The value of α and β are standard value for topic modeling as per extracted from previous literature. The iterations 1000 are the minimum number from which proper topic classification was observed. The Gibbs sampling process converges eventually and finding the optimal number of iterations remains as future work. The assignment of number of topics is a tricky bit; assigning high number can result in a very fine dissection of document into topics in which each word may represent different topic and a

low number will not be able to separate out topics in very lesser iterations. Both of the case is undesirable, thus topics are set considering user requirement of number of topics and training time.

After completion of the inference process for all the 600 documents iterated 1000 times, the topic-word and topic-document distribution matrices thus resulted was used to generate the topic distribution of the first document from the corpus. The first document of the corpus is of *business* category.

The Fig shown above shows the distribution of the 20 different topics in the first document of the corpus. The x-axis shows the topic number and the y-axis shows the probability of the topic in the document. From the figure we can see that the primary topics of the document are topic number 3, 8, 15 and 19. I then extracted the top 5 words associated with each of these topics from the document. These are shown in the list below.

1. bank (86) economy (66) spending (66) federal (66) banks (64)→**topic 3**
2. percent (231) year (220) china (123) prices (108) report (75)→**topic 8**
3. government (99) public (64) rules (52) make (46) plan (38)→**topic 15**
4. time (56) it's (45) years (44) return (42) put (37)→**topic 19**

From the manual evaluation of the words in the above topics it can be clearly seen that the above words relate to a *business* category. Since the highest ranking topics is of business category thus we can consider the document as of *business* category a coarse topic assignment. This proves the correctness of the model as applied in the research.

Further, it can be seen that even though all four topics are related to business they are slightly different in nature. Also this difference is across the topics but shows similarity within the topic. For example, topic 3 is more related to *banks* whereas topic 15 is more related to *government policy*. Similarly topic 8 and 15 also show different nature as compared to other topic. This shows the fine-grained topic assignment capability of the algorithm.

3.10 Conclusion

In this chapter the LDA algorithm is explained. Both the mathematical and the graphical representation of the algorithm is clearly pointed out and described. This chapter also

presents the various filters adopted to make the dataset clean and usable for the text classification process. Finally the chapter shows the implementation of LDA and presents the quantitative evaluation of the model thus created. The evaluation showed that not only the model can classify document into different categories but also extract the multiple sub-topics present within the document.

Chapter 4

TOPIC LABELING

4.1 Introduction

As the volume of available electronic information is rapidly increasing with the advancements in digital processing, the need for efficient techniques to organize those massive amounts of textual data becomes significant. One of the popular approaches for organizing such data is to implement clustering algorithms which can classify a set of documents into coherent clusters. The documents categorized under each cluster should be as similar as possible and those documents should be dissimilar from documents in other clusters.

In user interfaced applications, where human users directly interact with the created clusters, the clusters need to be labeled so that users can understand what the cluster is about. A popular approach for cluster labeling is to apply statistical techniques for feature selection that can be done by identifying important terms in the text that represent the cluster topic. In many cases list of significant keywords and phrases fail to provide a meaningful label for the documents and sometimes even though the selected terms are co-related they represent different aspect of the topic underlying the cluster. Thus, user's intervention is required to infer an appropriate label from the suggested terms to successfully describe the cluster's topic.

In my thesis, the contribution of external knowledge bases for cluster labeling is investigated. From the given cluster of documents important terms (keywords and phrases) are extracted from the text. Then a list of related Wikipedia pages are identified by searching Wikipedia using a query that is based on those significant terms. The Wikipedia categories and titles of related pages act as potential candidate for cluster labeling.

.

4.2 General Framework for Cluster Labeling

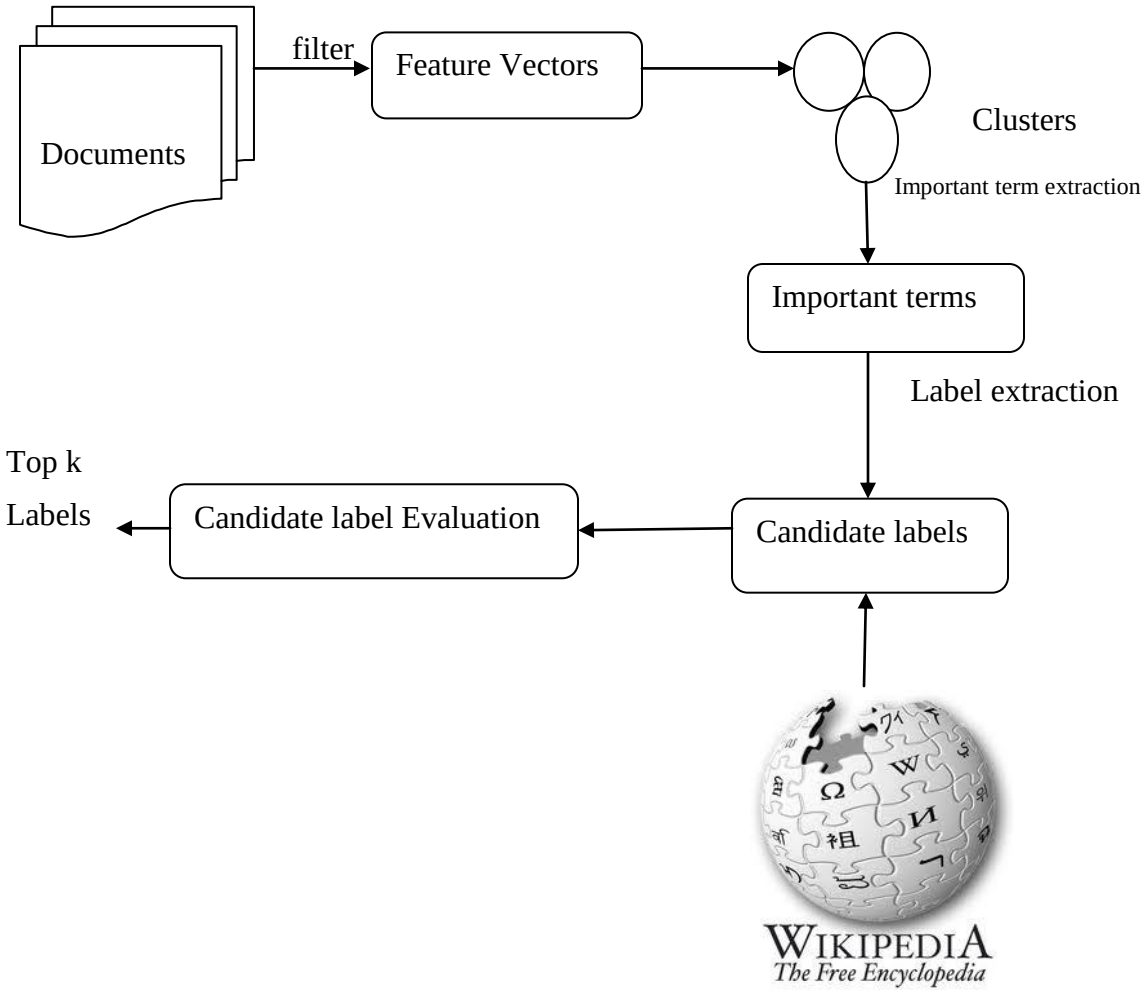

Fig 4.1: A general Framework for Cluster Labeling

The model receives a set of textual documents as input. Stemming is performed by removing the least important terms using various filters like word count, tf-idf and POS tags, and also the terms are derived into their root form. In this way each term in the documents are converted into feature vectors. Then, the documents are clustered using LDA. For generated clusters, a set of important terms are extracted to best represent the contents of the documents of the cluster. Those important terms of the cluster are used to identify a list of candidate labels for the cluster. Candidate labels are selected from the set of important terms from the clusters or any external sources (Wikipedia in my research).

4.2.1 Label Extraction

Basically candidate labels are extracted from two different types of sources. The first type involves labels that are extracted directly from the cluster's documents contents. Sometimes there are many cases in which important terms from the documents do not provide suitable labels or are not meaningful enough for end-users. Thus another external sources (Wikipedia in my case) from which candidate cluster labels are extracted is significant. The main reason for focusing on Wikipedia is its interactive ability to provide high quality controlled content. Moreover, Wikipedia content has also been annotated by Wikipedia's users. These manual annotations can provide high quality meaningful labels.

To extract the label from the Wikipedia pages, as an input, the clusters *C,* its significant terms T(C) and the collection of Wikipedia pages are taken. Then the semantic distance between T(C) and all the pages from the Wikipedia is calculated. The calculated semantic distance is ranked according to the ascending order i.e. lesser the distance higher the position it gets, where 1 being the highest position. The calculation is done for each cluster and the label is assigned as the title of the top ranking(lowest semantic distance) Wikipedia page. The above mentioned process involves two important concepts. First is the feature representation of both the T(C) and Wikipedia pages and second is the similarity distance calculation. For feature representation vector space model is used and similarity distance is measured using the cosine similarity.

4.2.1.1 Feature representation using vector space model

Vector space model (or term vector model) is an algebraic model for representing text documents (and any objects, in general) as vectors of identifiers. The identifier represents the content of the documents, for example a document can be represented as the count of words present in it. Vector space model is used in information filtering, information retrieval, indexing and relevancy rankings.

In my system there are two sets of inputs. First is the T(C), the important terms for the clusters and next are the Wikipedia pages. In vector space model T(C) is represented as the vectors of the words and each Wikipedia document is represented as the vector of words and their counts.

$T(C) = \{w_{1,t}\, w_{2,t} \ldots\ldots\ldots, w_{i,t}\}$

$D_j = \{\{w_{1,j}\, w_{2,j} \ldots\ldots\ldots, w_{i,j}\}\}$ where,

D_j represents the j^{th} Wikipedia page.

4.2.1.2 Semantic distance calculation using cosine similarity

The term semantic distance in this context represents how closely the meaning of one document is represented by the other. Thus the intuition behind assigning labels to the clusters generated by LDA; by leveraging the Wikipedia pages is that: if the documents share similar meaning then their title must also be similar.

The calculation is done by comparing the deviation of angles between each document vector and the original query vector where the query is represented as same kind of vector as the documents. In practice, it is easier to calculate the cosine of the angle between the vectors, instead of the angle itself:

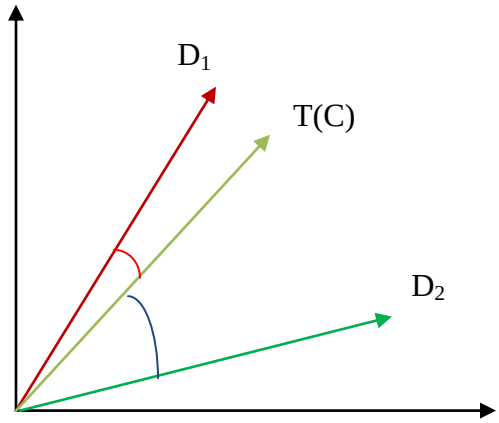

Fig 4.2: Angle distance between T(C) and two other documents

Fig shows 3 different vectors D_1, D_2 and T(C) and the angle calculation between them. From the figure it can be seen that T(C) is closer to D_1 than D_2. This can be interpreted as: the meaning of T(C) is closer to the document D_1 than document D_2. Thus, the title of the document D_2can be selected as the label for T(C).

The distance calculation here is done by the following formula: T(C) from here is represented as q.

$$score(q,d) = coord(q,d) * queryNorm(q) * \sum_{t\ in\ q}(tf(t\ in\ d) * idf(t)^2) \quad \text{equation (4.1)}$$

Where,

1. tf(t in d) correlates to the term's frequency, defined as the number of times term t appears in the currently scored document d. Documents that have more occurrences of a given term receive a higher score. The computation for tf(t in d) is:

$$tf(t\ in\ d) = \sqrt{frequency} \qquad \text{equation (4.2)}$$

2. idf(t) stands for Inverse Document Frequency. This value correlates to the inverse of document frequency (the number of documents in which the term t appears). This means rarer terms give higher contribution to the total score. The default computation for idf(t)is:

$$idf(t) = 1 + \log(\frac{number\ of\ documents}{document\ frequency+1} \qquad \text{equation (4.3)}$$

3. coord(q,d) is a score factor based on how many of the query terms are found in the specified document. Typically, a document that contains more of the query's terms will receive a higher score than another document with fewer query terms.
4. queryNorm(q) is a normalizing factor used to make scores between queries comparable. This factor does not affect document ranking (since all ranked documents are multiplied by the same factor), but rather just attempts to make scores from different queries (or even different indexes) comparable. The computation is:

$$queryNorm(q) = \frac{1}{sum\ of\ squared\ weights} \qquad \text{equation (4.4)}$$

Where,

$$sum\ of\ sqaured\ weights = \sum_{t\ in\ q} idf(t) \qquad \text{equation (4.5)}$$

4.3 Experiments

4.3.1 Data Collection

The clusters are generated from the same dataset as described in the previous chapter. LDA is used for clustering, which results in the cluster distribution of documents into 20 different topics. The Wikipedia documents are collected by downloading the Wikipedia dump[5]. The

[5]http://dumps.wikimedia.org/enwiki/20091103/

[6]http://lucene.apache.org/java/docs/index.html

[7]The annotation should have been done by more than one annotator and agreement score should have been

size of the Wikipedia dump is quite large (~7 GB). All the unwanted content (anything other than page body) is removed from the dump and indexed by Lucene[6]. This resulted in the index file of size ~1.3 GB.

4.3.2 Evaluation and Experimental setup

Table 4.1: Top 40 words for different topics

Topic 3	Topic 4	Topic 7
study cancer patients women found people researchers research vaccine percent health risk medical men drug published journal tobacco university disease hot products prostate brain fda drugs activity reported flashes nichols process school gene survival heart exercise week blood results cocaine	apple ipad android iphone tablet market device microsoft company mobile tablets windows devices software year at&t operating apple's display announced motorola phones technology phone launch verizon smartphones million intel galaxy samsung version consumers based electronics consumer wireless system features research	big texas network year constellation conference oklahoma news scott university college schools television espn state sports edf model agreement national programming business saturday events added east athletics pac meet received president home related klinsmann regional programs exposure partners high academic

First we extracted T(C) or q using LDA. Table 4 shows the top 40 words extracted for 3 different topics by LDA. These words generated for different topics are used to measure the similarity between Wikipedia documents indexed by lucene. Then we generate top 10 labels for each topic. Table 5 shows the label extracted for topic 4, 5 and 7. If the content of the Table 4 and **Table** 5are matched it can be seen that the labels extracted are highly co-related.

[6]http://lucene.apache.org/java/docs/index.html

Table 4.2: Top 10 labels generated by calculating semantic distance

Topic 3	Topic 4	Topic 7
Green tea	Open Handset Alliance	ESPN2
Oklahoma Medical Research Foundation	Verizon Wireless	College football on television
Epidemiology and etiology of breast cancer	Samsung Telecommunications	Criticism of ESPN
Tobacco smoking	Motorola Q	ESPNU
Effects of cannabis	Mobile ESPN	EDF Energy
Alcohol and cancer	BlackBerry	Insight Bowl
Cancer	Samsung Group	2006 Oklahoma Sooners football team
U.S. Food and Drug Administration	Motorola RAZR V3	Men's college basketball on television
Breast cancer	Symbian OS	College GameDay (football)
Prostate cancer	Qwest Wireless	Fiesta Bowl

The order of the label shown above in the table 5 are ranked according to the similarity score calculated i.e. the label at the top is the most relevant label calculated by the system. To evaluate this, the entire extracted labels are ranked manually. The ranking is done according to the relevance of T(C) to the extracted label. This ranked label is considered as the gold standard[7].

[7]The annotation should have been done by more than one annotator and agreement score should have been calculated but this was not possible due to time constraint.

Table 4.3: Extract taken from the gold standard

Topic 3	Topic 4	Topic 7
Green tea =10 Oklahoma Medical Research Foundation =1 Epidemiology and etiology of breast cancer =2 Tobacco smoking =3 Effects of cannabis =9 Alcohol and cancer =5 Cancer =6 U.S. Food and Drug Administration =8 Breast cancer =4 Prostate cancer =7	Open Handset Alliance =1 Verizon Wireless =9 Samsung Telecommunications =3 Motorola Q =5 Mobile ESPN =2 BlackBerry =6 Samsung Group =4 Motorola RAZR V3 =7 Symbian OS =8 Qwest Wireless =10	ESPN2 =2 College football on television =1 Criticism of ESPN =4 ESPNU =3 EDF Energy =10 Insight Bowl =8 2006 Oklahoma Sooners football team =5 Men's college basketball on television =6 College GameDay (football) =7 Fiesta Bowl =9

Next for the evaluation two different measures are used. First the accuracy of the list is calculated. This is done to measure the correctness of the label extraction method. The label extracted is deemed accurate if the top ranked label from the gold standard is present in the top 5 position of the list of label extracted by the system. For example, in the Table 6 which contains the manual label for the extracted labels, the rank 1 is present within the top 5 label for the topic. Rank 1 is present in position 2^{nd}, 1^{st} and 2^{nd} of the topic 3, 4 and 7 respectively. Thus for these topics the system has extracted the correct label. The evaluation showed that the out of 20 topics 15 of the topics had the label correctly identified by the system. Thus the accuracy of the system is 75%. Even though this is not very high accuracy the evaluations measure used was very strict since top 5 is quite a restrictive choice. Judging by the extracted labels none is irrelevant to the words in the topics. This proves that the labels extracted were indeed highly correlated to the extracted topics. Also the documents that were used for clustering were from broad categories Sports, Business, Science and technology and Health. These are very broad categories but by using the model fine-grained categories are extracted successfully within these broad categories. For example, if the contents of Topic 3 are observed from above tables, it can be seen that this came from the documents of category health. The model extracted the label for it as ``green tea'', which highly relates to health but

is very specific to a fine category. Also topic 4 came from Science and technology, and the system specifically categorized it as “Open Handset Alliance” and for topic 7 which came from sports the system identified it as a text talking about college football. This shows the successful implementation of the model.

Table 4.4: Pearson Correlation coefficient for extracted label

Topic	**Pearson Correlation Coefficient**
1	0.73
2	0.68
3	0.91
4	0.85
5	0.65
6	0.55
7	0.73
8	0.74
9	0.63
10	0.55
11	0.44
12	0.96
13	0.55
14	0.71
15	0.75
16	0.77
17	0.64
18	0.62
19	0.83
20	0.81

The next evaluation measure used is to calculate the Pearson correlation coefficient. This is a necessary evaluation as this measures the correctness of the ranking generated by the system against the gold standard. If the ranking generated by the system completely matches the ranking on the gold standard then its Pearson correlation coefficient will be 1. The value of the coefficient decreases with each mismatch. In such a small list absolute match of each

element is very hard so to get a fair evaluation the matching is done within the window of 1 index at the top and 1 at the bottom. Thus a match is considered if the index of gold standard is same as the generated index or if it is within index-1 or index+1. It can be seen from Table 7 that the correlation coefficient shows acceptable result. This proves that the ranking generated by the system is statistically similar to the ranking generated by human annotator.

4.4 Conclusion

In this chapter a method to assign label to the clusters generated by using LDA is implemented successfully. First it is shown using the terms in the clusters are not very applicable for the label of the clusters. Then a method is implemented that leveraged the Wikipedia knowledge resource to calculate the semantic distance between the clusters and the Wikipedia documents to generate a ranked list of labels for the clusters. The evaluation showed promising and accurate results for the cluster label. The method has become successful to generate very fine grained and contextually significant labels for each cluster.

Chapter 5

CONCLUSION

5.1 Summary of results and contributions

The main aim of this research was to implement an unsupervised learning approach to utilize unlabelled data for document classification. The research successfully accomplished the unsupervised learning task.

The research explained that any document may comprise of more than one topics, thus labeling all the topics within a single document manually is very hard and time consuming task. For this reason unsupervised algorithm becomes the best approach to take. The aim then is to associate each word within a document to respective topics and thus create topic clusters (distribution) of the document.

The research showed the generative model that a document creation process follows. An algorithmic and graphical representation (plate diagram) of the generative model for document generation is well explained in this research. This property shown by the document subsequently led to use of probabilistic models well suited for generative process.

The research explained a very effective method for unsupervised learning; LDA which in conjunction with Gibbs Sampling is an efficient algorithm to extract topics form documents. LDA is one of the most computationally easy and tractable approach to topic modeling then other model likes PLSA, LSA etc.

The research was successful to determine the topic of the given document. Determination of the topic of the given document was done through the statistical solution to the problem of managing large archives of documents. The added advantage of using probabilistic model like LDA is that it assigns a topic to a word considering the association of that word with other word in the document and also document topic association. For example word ‘bank’ would be assigned to a topic “nature” or “business” depending upon the number of times each topic has been assigned to the document and the number of words in that document that has been assigned to particular topic. This helps to disambiguate semantic meaning of the word.

5.2 Future Work Direction

Although the research showed some very promising results but it also has some short comings. The major short coming is the limited data set. The data set size is quite small, and we could see that the accuracy (quantitative; mainly human judgment) obtained by the unsupervised classifier in the small data set is not extraordinary. We could not test the full potential of the method due to the lack of abundance data. Although many free datasets are available, non are under the topics that this research wants to extract (sports, business, science and technology and health). Collection of data over the research duration by a single person is not resulting on the amount required. Once proper amount of data is collected full evaluation of the models can be conducted.

For now the generated clusters are labeled observing the semantic distance between significant terms under each cluster and terms obtained from an external source (Wikipedia). Future directions for this can be the implementation of various judges like Mutual Information judge and Score Propagation judge that not only use the terms from clusters and an external source like Wikipedia for label evaluation but also another external source from web that is corpus containing trillions of words that can enhance the labeling.

REFERENCES

[1] F. Sebastiani, "Machine learning in automated text categorization," vol. 34, no. 0360-0300, 2002.

[2] P. E. H. D. G. S. Richard o. Duda, *Unsupervised Learning and Clustering, Ch. 10 in Pattern classification (2nd edition), p. 571*. New York: ISBN 0-471-05669-3, 2001.

[3] N. Bharati, "Semi-supervised classification of news documents," 2011.

[4] T. Joachims, "Text categorisation with suport vector machines: learnign with many relevant features," in *Proceedings of ECML-98, 10th European Conference on Machine Learning*, 1998, pp. 137-142.

[5] G. Adami, P. Avesani, and D. Sona, " Clustering documents in a web directo," in *Proceedings of the 5th ACM international workshop on Web information and data management*, 2003, pp. 66-73.

[6] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent Dirichlet allocation," *Journal of Machine Learning Research*, pp. 993-1022, 2003.

[7] T. L. Griffiths and M. Steyvers, "Finding scientific topics," in *Proceedings of the National Academy of Sciences*, 2004, pp. 5228-5235.

[8] T. Hoffman, "Probabilistic Latent Semantic Analysis," in *Uncertainty in Artificial Intelligence*, 1996.

[9] D. Blie, A. Ng, and M. Jordan, "Latent Dirichlet Allocation," in *Journal of machine Learning Research*, 2003, pp. 992-1023.

[10] F. Sebastiani, "Machine earning in sutomated text categorisation," in , 2002, p. vol34,no,03600300.

[11] Y. a. Pedersen, "A comparative Study on Feature Selection in Text Categorisation," in *proceedings of the Fourteenth International Conference on Amchine Learning*, 1997, pp. 412-420.

[12] H. L. a. H. Motoda, "Feature Selection for Knowledge Discovery and Data Mining," 1998.

[13] H. T.K, " Fast Identification of Stop Words for Font Learning and Keyword Spotting," in *Proceedings of the Fifth International Conference on Document Analysis and Recognition*, 1999, pp. 333-336.

[14] J. W. a. K. Sorotkin, "The automatic identification of stop words," *Journal of Information Science*, pp. 45-55, 1992.

[15] D. A. Hull, ". Stemming algorithms: a case study for detailed evaluation," *Journal of the American Society for Information Science*, pp. 70-84, 1996.

[16] R. Braschler, "How Effective is Stemming and Decompounding for German Text Retrieval?," in *Information Retrieval*, 2004, pp. 291-316.

[17] H. T. ,. G. W. B. &. L. K. L. Ng, "Feature selection, perception learning, and a usability case study for text categorization," in *Proceedings of the 20th annual international ACM SIGIR conference on Research and development in information retrieval*, 1997, pp. 67-73.

[18] G. Wang and F. H. Lochovsky, " Feature selection with conditional mutual information maximin in text categorization," in *Proceedings of the thirteenth ACM international conference on Information and knowledge management*, 2004, pp. 342-349.

[19] C. Chen, H. Lee, and C. Hwang, "A Hierarchical Neural Network Document Classifier with Linguistic Feature Selection," in *Applied Intelligence*, 2005, pp. 277-294.

[20] B. Zang, et al., "Intelligent GP fusion from multiple sources for text classification," in *Proceedings of the 14th ACM international conference on Information and knowledge*

management, 2005, pp. 477-484.

[21] E. Riloff, " Little words can make a big difference for text classification," in *Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval*, 1995, pp. 130-136.

[22] J. S. Youngjoong Ko, "Automatic Text Categorization by Unsupervised Learning," in , vol. Volume 1, Saarbrücken, Germany, 2000.

[23] Y. Y. a. J. O. Pedersen, "A comparative study on feature selection in text categorization," in , Nashville, US, 1997.

[24] T. Joachims, "Text categorization with support vector machines: learning with many relevant features," *Proceedings of ECML-98, 10th European Conference on Machine Learning*, pp. 137--142, 1998.