# DOCUMENT SUMMARIZATION USING AUTOMATICALLY EXTRACTED KEYPHRASES

*dissertation submitted to*
*Jawaharlal Nehru University*
*in partial fulfillment of the requirement*
*for the award of the degree of*

**MASTER OF TECHNOLOGY**

in

**COMPUTER SCIENCE & TECHNOLOGY**

By

Suma Boddu

SCHOOL OF COMPUTER AND SYSTEMS SCIENCES
JAWAHARLAL NEHRU UNIVERSITY
NEW DELHI – 110067, INDIA
JULY 2005

SCHOOL OF COMPUTER & SYSTEMS SCIENCES
JAWAHARLAL NEHRU UNIVERSITY
NEW DELHI – 110067 (INDIA)

## CERTIFICATE

This is to certify that the thesis entitled **"DOCUMENT SUMMARIZATION USING AUTOMATICALLY EXTRACTED KEYPHRASES"**, being submitted by Miss Suma Boddu to the **School of Computer & Systems Sciences, Jawaharlal Nehru University** in partial fulfillment of the requirement for the award of the degree of **Master of Technology**, is a record of original work done by her under the supervision of Prof. K. K. Bharadwaj, during the winter semester, 2005.

The results reported in this thesis have not been submitted in part or full to any other University or Institution for the award of any degree etc.

B. Suma
Suma Boddu
(Student)

[illegible] 11-07-05
Prof. K.K. Bharadwaj
(Supervisor)

[illegible] 8/5/08
(Dean)
SC & SS
J.N.U, New Delhi – 110067

# ACKNOWLEDGEMENT

With a deep sense of gratitude, I wish to express my sincere thanks to my guide/supervisor, **Prof. K. K. Bharadwaj,** for his immense help in planning and executing this master's dissertation work in time. He consistently stood by me in all my difficult times, helping me to do my research fruitfully, giving valuable suggestions. In all respects, I am grateful to him for the time he has spent with me in discussions, or finding out the ways for me whenever I was struck in understanding things by not only guiding me to find the way but also by searching all relevant material for me. It would be impossible for me to come out successfully without his constant guidance.

I would also pay my gratitude to Prof. Karmeshu (Dean, SC & SS) for being a source of inspiration all throughout as well as to my faculty members of SC & SS for their support.

I cannot find the words to express my profound feeling of gratitude for my parents, my father Sri B.Babu Rao and my mother Smt. B.Koteswaramma, who taught me the value of hard work by their own example. I would like to share this moment of happiness with them. They rendered me enormous support during the whole tenure of my education.

There are special mentors that I acknowledge due to their importance in my work. I believe it's appropriate to acknowledge all of these. Since I can't name all of them, I owe my debt and extremely warm gratitude to all of whose honorable references (as well as those whom they have taken reference of), I have referred to in my reference section while working on the dissertation.

Last but not least I would like to thank my classmates, friends for being supportive and helping me in need for completing my dissertation.

**Suma Boddu**

# Abstract

With tons of information pouring in everyday, text summaries are becoming essential. The goal of text summarization is to take a textual document, extract content from it and present the most important content to the user in a condensed form and in a manner sensitive to the user's needs. Instead of having to go through the entire text, people can understand the text quickly and easily by means of a concise summary. The title, abstract and keywords if provided can convey the main ideas, but they are not always present in a document. The summarization can either be an extract consisting entirely of material copied from the input, or an abstract containing material not present in the input. In order to generate a summary, we have to identify the most important pieces of information from the document, omitting irrelevant information and minimizing details, and assemble them into a compact coherent report. These important pieces of information are the keyphrases for the document. A keyphrase for any given text document is a set of one to three words which appear consecutively in the text and captures main topic. Keyphrases are useful for a variety of purposes, including summarizing, indexing, labeling, categorization, clustering, highlighting, browsing and searching. The present thesis focuses on summarization of text documents using automatically extracted keyphrases. Kea algorithm, which is based on the naive bayes scheme, is used for automatic extraction of keyphrases. Using these keyphrases the most important sentences from the document can be selected to give the summary for that document. The summary is generated by direct selection of sentences and also by using mutual reinforcement principle. Further a sentence ordering scheme based on cohesion measure among the sentences is applied to order sentences in the generated summary. For a brief summary top most sentences can be extracted. The proposed scheme of summarization is demonstrated through experimental results.

**dedicated to…**

*my beloved parents.*

# CONTENTS

# Chapter 1

# Introduction

Knowledge Discovery in Databases (KDD) has been attracting a significant amount of research and industry attention in recent years. The major reason that KDD has attracted such a great deal of attention is due to the availability of huge amount of data and the imminent need for turning such data into useful information and knowledge. In simple words, KDD can be defined as the automatic extraction of invisible and hidden knowledge from large volumes of data.
**KDD is the non-trivial process of identifying valid, novel, potentially useful, and ultimately understandable patterns in data [6].**

KDD includes data cleaning, data integration, data selection, data transformation, data mining, pattern evaluation and knowledge presentation. KDD application areas include marketing, finance, fraud detection, manufacturing, telecommunications and Internet agents. Historically, the process of extracting knowledge from large amounts of data is known as Data Mining. Many people treat KDD and Data Mining as synonyms, while others view Data mining as an essential step in the process of KDD. But the term Data Mining has gained most popularity in information industry. KDD also called as Data Mining is typically conducted on structured, relational databases.

Text mining has extended the applicability of KDD dramatically by the use of sophisticated natural language processing. Text mining is defined as a special form of data mining, which is applied to large volumes of non-structured text files instead of to numerical, structured data.

**Text mining is a branch of data mining. Its objective is to analyze the texts of complete text collection in order to select relevant texts or extracts, to categorize**

**texts, and to give overviews of the text collection according to the user's interest [15].**

Different results from text mining can be distinguished: texts from the collection are selected according to the user's interest, extracts taken from a text may be presented to the user or overviews of the texts are given as shown in Fig 1.

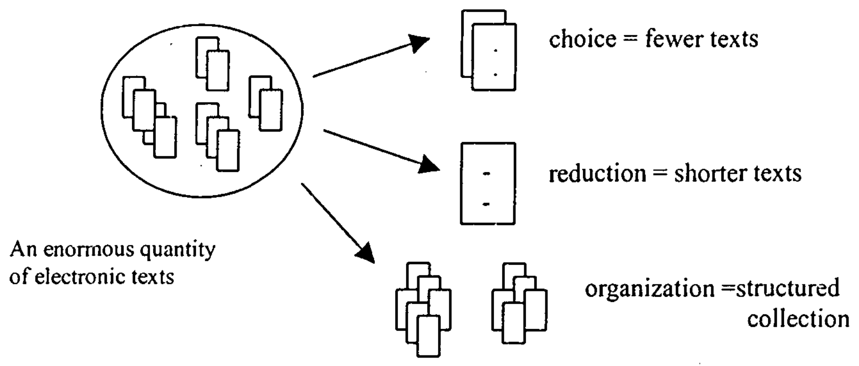
choice = fewer texts
reduction = shorter texts
An enormous quantity of electronic texts
organization =structured collection

Fig 1: Text mining objectives

The amount of text database is enormously increasing day by day. There is a great need to extract special information that can only be found by digging out the huge database. For this reason, text mining is becoming more and more important to enable users to turn volumes of data into new information that is useful for variety of purposes.

Some typical Text mining tasks can be characterized as follows:

- Information Retrieval

  Retrieval of textual documents.

- Information Extraction/Keyphrase Extraction

  Extraction of partial knowledge from the text.

- Analysis of text collection

  Providing an overview of text collection. Categorization, Clustering, Classification and Summarization tasks belong to this group.

We briefly discuss about the importance of keyphrases and the need for automatic keyphrase extraction algorithms, summarization and the need for automatic summarization techniques in the following sections.

## 1.1 Keyphrase Extraction

Keyphrase extraction is one of the tasks of Text Mining. The main goal of keyphrases is to provide the user with the overview of documents' contents. Thus with the help of keyphrases the user can decide whether or not a document is relevant to him/her. Keyphrases are useful for a variety of purposes, including summarizing, indexing, labeling, categorization, clustering, highlighting, browsing and searching. Keyphrases are usually chosen manually. However, with the amount of text flooding in recent years it has become very tedious to assign keyphrases manually. Considering the fact that it is hard and time consuming to manually assign keyphrases to documents, automatic keyphrase extraction algorithms [5, 7, 23, 25] have been developed using artificial intelligence and natural language processing (NLP) techniques.

The goal in keyphrase extraction is to produce topical and most indicative phrases, for any type of factual document [23]. The task of automatic keyphrase extraction is to select keyphrases from within the text of a given document. Automatic Keyphrase Extraction makes it feasible to generate keyphrases for the huge number of documents that do not have manually assigned keyphrases. Automatic Keyphrase Extraction is a special case of more general task called *keyphrase generation*, which tries to find keyphrases that may or may not be in the document. Obviously, keyphrase generation is a harder problem in the sense that machine understanding of the document is needed.

## 1.2 Summarization

With the rapid growth of the World Wide Web and electronic information services, information is becoming available on-line at an incredible rate. No one has time to read everything, yet we often have to make critical decisions based on what we are able to absorb. In order to fully utilize these on-line documents effectively, it is crucial to be able to extract the gist of these documents. Summarization, the art of extracting key content from one or more information sources, has become an integral part of everyday life. People keep abreast of world affairs by listening to news bites. They go to movies largely on the basis of reviews. With summaries, they can make effective decisions in less time.

Summarization is the process of condensing a source text into a shorter version preserving its information content. Its goal is to include in that summary the most important facts in the document. Summarization serves the purpose of indicating what a given text is about. A good summary will tell a reader whether he or she wants to read the whole document. A wide variety of texts can benefit from summarization, including newspapers and journals, press releases, scientific reports and organizational memos.

In most cases, summaries are written by human, but nowadays, due to the overwhelming quantity of information and the need to access the essential content of documents accurately to satisfy users' demands, calls for the development of computer programs able to produce text summaries. In order to generate a summary, we have to identify the most important pieces of information from the document, omitting irrelevant information and minimizing details, and assemble them into a compact report. The important pieces of information from the document are called the keyphrases and one of the methods to generate summary is using these keyphrases.

## 1.3 Problem Specification

This dissertation describes one possible approach for document summarization using automatically extracted keyphrases. The present work includes generation of keyphrases from any given factual document and using these extracted keyphrases to generate documents' summary. Keyphrase Extraction algorithm, KEA [7, 25] is used to extract keyphrases from a given text document. Using these keyphrases, summary is generated for the given document. The summary can be generated by direct extraction of sentences, which contain the most important keyphrases [19], and also by using Mutual reinforcement principle [14]. We have used both the methods to generate summary and the results obtained by both the methods are compared to judge the better summarization technique. Further, a sentence-ordering scheme based on cohesion measure among the sentences is applied to order the sentences in the extracted summary. The basic idea behind the extraction of keyphrases and then summarizing the document is, as discussed above keyphrases give the overview of documents' contents. So using these keyphrases, which give the essence of a given document, the sentences that contain these keyphrases can give a meaningful summary in few lines about the document, which makes it easier for the reader to know what the document is about and hence saving the time to read the whole document.

## 1.4 Outline of the Dissertation

This thesis is organized as follows: Chapter 2 depicts the need and importance of keyphrases and some well-known techniques for keyphrase extraction. This section discusses the need for automating the process of keyphrase extraction. It also briefly discusses various existing algorithms for this purpose. Chapter 3 comprises of an explanation for the demand in the area of text processing and in particular, about the need for text summarization. It specifies the great need for automatic summarization techniques and their applications. This section also gives a brief explanation about the already existing approaches for automatic text summarization. Chapter 4 consists of the main work done as a part of dissertation. This section explains in detail the procedure for summarization using automatically extracted keyphrases, which is the main intention of the present work. It discusses distinct methods to summarization using keyphrases and also the technique for ordering of extracted sentences in the summary obtained. Chapter 5 deals with the implementation details and the experimental results. Chapter 6 discusses the conclusion and future enhancements for the present work

# Chapter 2

# Keyphrase Extraction

A keyphrase for any given text document is a set of one to three words which appear consecutively in the text and captures main topic. Keyphrases give a very short summary of a document, which makes it easier for the readers to determine if the document is relevant to the their information needs.

Keyphrases are meant to serve several goals:

- when printed on the first page of the journal, the goal is summarization.
- when printed in the cumulative index of the journal, the goal is indexing.
- when the search engine field contains the field labeled *keywords*, the goal is to make the search more precise.

Though keyphrases are very useful, only small minorities of documents have keyphrases assigned to them, and manually assigning keyphrases to existing documents is quite tedious and costly. Therefore, there is a need for automatic keyphrase extraction algorithms. It is very difficult to design computers capable of understanding the meaning of human language. Hence, instead of making the computers understand content of the document, statistical methods are used to extract important topical phrases (i.e. keyphrases) from the document.

Keyphrase extraction is a classification task: a document can be seen as a set of phrases, and a keyphrase extraction algorithm should correctly classify a phrase as a keyphrase or a non-keyphrase. Machine learning techniques can automate this task if they are provided with a set of training data: examples of both keyphrases and non-keyphrases. The data are used to teach the algorithm how to distinguish keyphrases from non-keyphrases. The resulting algorithm can then be applied to new documents for keyphrase extraction.

## 2.1 Different Approaches for Generating Keyphrases

The task of automatically generating keyphrases for the documents has two fundamentally different approaches: *Keyphrase assignment* and *Keyphrase extraction*. Both approaches use supervised machine learning from examples. In both the cases the training examples are documents with manually assigned keyphrases.

### *2.1.1 Keyphrase Assignment*

Keyphrase assignment seeks to select the phrases from a controlled vocabulary (a predefined list of keyphrases) that best describe a document [25]. In keyphrase assignment, there is a predefined list of keyphrases (*controlled vocabulary*). A document is converted to a vector of features and machine learning techniques are used to induce a mapping from the feature space to the list of keyphrases [24]. The features are based on the presence or absence of various words or phrases in the input documents. Usually, a document may belong to several different classes. That is, a learned model will map an input document to several different controlled vocabulary keyphrases.

### *2.1.2 Keyphrase Extraction*

In keyphrase extraction, keyphrases are selected from within the body of the input document, without a predefined list. This suggests the possibility of using author-assigned text keyphrases to train a keyphrase extraction system [24]. In this approach, a document is treated as a set of candidate phrases and the task is to classify each candidate phrase as either a keyphrase or non-keyphrase. A feature vector is calculated for each candidate phrase and machine learning techniques are used to learn a model that can classify a phrase as a keyphrase or non-keyphrase. The features include the frequency and location of the candidate phrase in the input document.

A learning algorithm is *training-intensive* when it requires a relatively large amount of labeled training examples in order to perform well [24]. Keyphrase assignment is training-intensive when the controlled vocabulary is large, since there must be several training example documents for each keyphrase in the vocabulary. On the other hand, keyphrase extraction typically works well with only about 50 training documents. A learning algorithm is *domain-specific* when the learned model does not generalize well from one domain to another domain. Keyphrase assignment is domain-specific, since the appropriate controlled vocabulary will vary from one domain to another. For example, the vocabulary of physics articles is distinct from the vocabulary of computer science articles. On the other hand, keyphrase extraction performs well when trained on articles from one domain and then tested on articles from a completely different domain.

## 2.2 Keyphrase Extraction Algorithms Studied

There exist many approaches for automatic keyphrase extraction. Few of the keyphrase extraction methods namely GenEx [22, 23], KEA [7, 25], LAKE (Learning Algorithm for Keyphrase Extraction)[5] are briefly described below. GenEx and KEA are considered notable algorithms for keyphrase extraction.

### 2.2.1 GenEx: A Hybrid Genetic Algorithm for Keyphrase Extraction

**GenEx** [22, 23] is a system developed by Peter Turney at NRC of Canada for automatic keyphrase extraction. GenEx is a hybrid of ***Genitor***, steady-state genetic algorithm and the ***Extractor***, parameterized keyphrase extraction algorithm. Extractor works by assigning a numerical score to the phrases in the input document. The final output of the Extractor is essentially a list of the highest scoring phrases. The behavior of scoring function is determined by a dozen numerical parameters. Genitor tunes the setting of these parameters, to maximize the performance of Extractor on the given training examples.

*2.2.1.1 Extractor*

Extractor takes a document as input and produces a list of keyphrases as output. Extractor has 12 parameters that determine how it processes the input text. Extractor algorithm is a ten-step process. The ten-step process of Extractor is depicted in Fig 2. The ten steps of the algorithm are as follows:

1. *Find Single Stems:* Get the list of all words in the input text. Drop words with less than three characters, drop all stop words (word such as "the", "and", "or", etc). Convert all the remaining words to lower case. Stem the words using Lovins or Porter stemming algorithm.

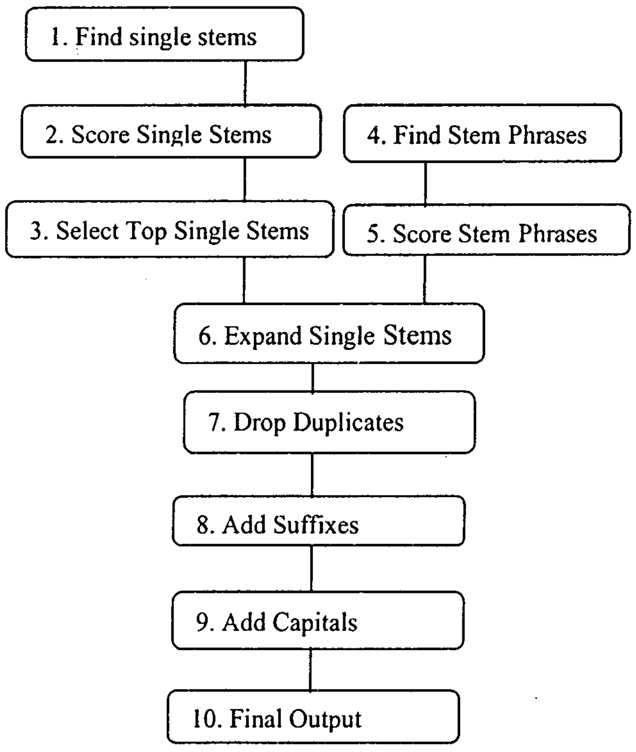

Fig 2: An overview of the Extractor algorithm

2. *Score Single Stems:* For each single stem, count the frequency of its occurrence in the input text and also note its first appearance.
3. *Select Top Single Stems:* Rank the stems in the decreasing order of their scores and make the list of top single terms.

4. *Find Stem Phrases:* Make the list of all stem phrases in the input text. A phrase is a sequence of one, two, or three words that appear consecutively in the text, with no intervening of stop words or punctuation. Stem each phrase using Lovins or Porter stemming algorithm.
5. *Score Stem Phrases:* For each stem phrase, count the frequency of its occurrence and also note its first appearance in the text.
6. *Expand Single Terms:* For each stem in the list of top single terms, find the highest scoring stem phrase of one, two, or three stems that contains the given single stem. Keep the list ordered by the scores calculated in Step 2.
7. *Drop Duplicates:* The list of top ranking phrases may contain duplicates. For example, two single stems may expand to the same two-word stem phrase. Delete duplicates from the ranked list of top ranking stem phrases, preserving the highest ranked phrase.
8. *Add Suffixes:* For each of the remaining stem phrases, find the most frequent corresponding whole phrase in the input text.
9. *Add Capitals:* For each whole phrase, find the best capitalization i.e. with the least number of capitals.
10. *Final Output:* Final result obtained will be an ordered list of mixed-case phrases with suffixes added. The list is ordered by scores calculated in step 2.

    Thus the output is the set of top ranking keyphrases which capture the main document.

#### *2.2.1.2 Genitor*

Genitor is used to tune the Extractor. Genitor genetic algorithm is used to maximize the performance (fitness) on training data by tuning the extractor parameters. The setting up of the dozen parameters, shown in Table 1, for Extractor is determined by a training process, during which Genitor searches through the parameter space for values that yield a high overlap between the keyphrases assigned by the authors and the phrases that are output by Extractor. After training, the best parameter values can be hardcoded in Extractor, and Genitor is no longer needed.

A genetic algorithm can be viewed as a method for optimizing a string of bits. A genetic algorithm works with a set of bit strings called *population* of *individuals*. The initial population is usually randomly generated. New individuals (new bit strings) are created by using *mutation* or *crossover* techniques. Mutation is the process of randomly changing existing individuals. Crossover is the process of combining substrings from parents to make new children. Each individual is assigned a score called *fitness* based on some measure of the quality of the bit string, with respect to a given task.

The twelve parameters of Extractor, with sample values

| Parameter Number | Parameter name | Sample value | Number of bits | Description |
|---|---|---|---|---|
| 1 | NUM_PHRASES | 10 | 0* | length of final phrase list |
| 2 | NUM_WORKING | 50 | 0+ | length of working list |
| 3 | FACTOR_TWO_ONE | 2.33 | 8 | factor for expanding to two words |
| 4 | FACTOR_THREE_ONE | 5.00 | 8 | factor for expanding to three words |
| 5 | MIN_LENGTH_LOW_RANK | 0.9 | 8 | low rank words must be longer |
| 6 | MIN_RANK_LOW_LENGTH | 5 | 5 | short words must rank higher than this |
| 7 | FIRST_LOW_THRESH | 40 | 10 | definition of early occurrence |
| 8 | FIRST_HIGH_THRESH | 400 | 15 | definition of late occurrence |
| 9 | FIRST_LOW_FACTOR | 2.0 | 8 | reward for early occurrence |
| 10 | FIRST_HIGH_FACTOR | 0.65 | 8 | penalty for late occurrence |
| 11 | STEM_LENGTH | 5 | 4 | max characters for fixed length stem |
| 12 | SUPRESS_POWER | 0 | 1 | flag for suppressing proper nouns |
|  | Total number of bits in binary string |  | 72 |  |

* This parameter is set by the user to the desired value.
+ This parameter is set to five times the NUM_PHRASES.

Table 1: The twelve parameters of Extractor with sample values

#### *2.2.1.3 GenEx*

The parameters in Extractor are set using the standard machine learning paradigm of supervised learning. The learning process involves adjusting the parameters to maximize the match between the output of Extractor and the target keyphrase lists, using the training data. The user sets the value for NUM_PHRASES and NUM_WORKING is set to five times NUM_PHRASES. The remaining ten parameters are set by the Genitor. Genitor uses a binary string of 72 bits to represent the ten parameters as shown in Fig 3. Each trial consists of running Extractor with the parameter settings specified by in the given binary string processing the entire training set. The fitness measure for the binary string is based on the average precision for the whole training set. Fitness is calculated using the following formulas:

total_matches = total number of matches between GenEx and human

total_machine_phrases = total number of phrases output by GenEx

precision = total_matches / total_machine_phrases

num_docs = number of documents in training set

total_desired = num_docs . NUM_PHRASES

penalty = (total_machine_phrases / total_desired )$^2$

fitness = precision . penalty

### 2.2.2 KEA

**KEA** [7, 25], a system developed by Witten, Frank et al at New Zealand Digital Library is another well-known approach for automatically extracting keyphrases. Keyphrase extraction is a classification task: each phrase in a document is either a keyphrase or not, and the problem is to correctly classify a phrase into one of the two categories. Kea uses Naive Bayes machine learning technique for classification. Kea's extraction algorithm consists of two stages. First, it creates a model for identifying keyphrases (using training documents where the author's keyphrases are known). Then, it applies the model to a new document.

Kea two step process includes training and extraction. In *training phase* a model is created to identify keyphrases using training documents where the author's keyphrases are already known. The next phase is the *extraction phase* where keyphrases are chosen from a new document using the model built during training. The Kea extraction process is depicted in Fig 3.

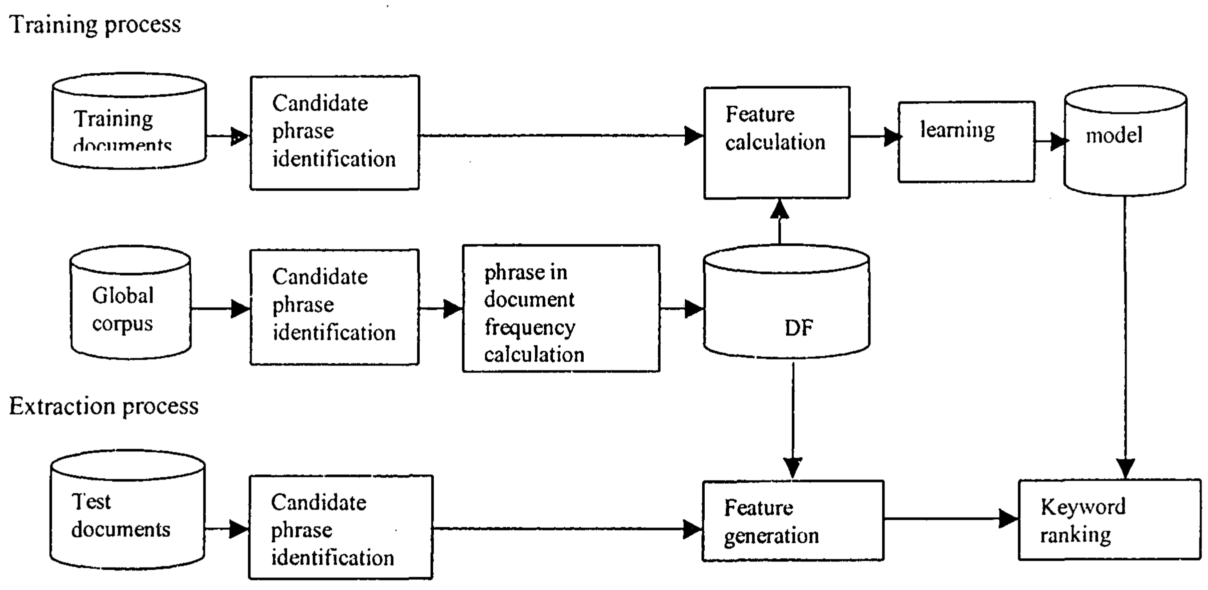

Fig 3: The training and extraction process in KEA

Kea identifies candidate keyphrases using lexical methods, calculates feature values for each candidate and uses a machine learning algorithm to predict which candidates are good keyphrases.

*2.2.2.1 Candidate phrases*

1. Input cleaning :

   The input stream is split into tokens and

   - Punctuation marks, brackets and numbers are replaced by phrase boundaries.
   - Apostrophes are removed
   - Hyphenated words are split into two.
   - Remaining non-token characters are deleted.

2. Phrase identification :

   - Candidate phrases are limited to certain maximum length.
   - Candidate phrases cannot be proper names.
   - Candidate phrases cannot begin or end with a stop-word (stop words consist of articles, conjunctions, adjectives, prepositions etc.).

3. Case-folding and stemming :

The final step in determining candidate phrases is to casefold all the words and stem them using Iterated Lovin's stemming algorithm.

*2.2.2.2 Feature Calculation*

Two features are calculated for each candidate phrase: TF x IDF *(term frequency x inverse document frequency)* and the first occurrence.

*TF x IDF*

This feature compares the frequency of the phrase's use in a particular document with the frequency of that phrase in general use.

$$\text{tf x idf} = \frac{\text{freq(P,D)}}{\text{size(D)}} \times -\log_2 (\text{df(P)}/ \text{N})$$

1. freq(P,D) is the number of times P occurs in D.
2. size(D) is the number of words in D.
3. df(P) is the number if documents containing P in the global corpus.
4. N is the size of global corpus.

*First occurrence*

The first occurrence is the number of words that precede the phrase's first appearance divided by the number of words in the document.

*Discretization*

Discretization is a process that transfers quantitative data into qualitative data. Both the features (tf x idf and first occurrence) are real numbers and must be converted to nominal data for the machine learning scheme.

*2.2.2.3 Training: building the model*

The training stage uses a set of training documents for which the author's keyphrases are known. For each training document, candidate phrases are identified and their feature values are calculated. Each phrase is then marked as keyphrase or a non-keyphrase, using the actual keyphrases for that document.

### *2.2.2.4 Naïve Bayes Classifier*

Bayes classifiers are statistical classifiers. They can predict the probability that a given sample belongs to a particular class. Bayesian classification is based on Bayes Theorem.

*Bayes Theorem:*

Let X be the data record (case) whose class label is unknown. Let H be some hypothesis, such as "data record X belongs to a specified class C." For classification, we need to determine P (H|X) (the probability that the hypothesis H holds), given the observed data record X. P (H|X) is the posterior probability of H conditioned on X. P(H) is the prior probability, or apriori probability, of H. The posterior probability, P (H|X), is based on more information (such as background knowledge) than the prior probability, P(H), which is independent of X. Similarly, P (X|H) is posterior probability of X conditioned on H. P(X) is the prior probability of X. Bayes theorem is useful in that it provides a way of calculating the posterior probability, P(H|X), from P(H), P(X), and P(X|H).

Bayes theorem [9]

$$P(H|X) = P(X|H) \, . \, P(H) / P(X)$$

Simple Bayesian classifier is Naïve Bayesian Classifier. Naïve Bayes Classification is a simple probabilistic classification method. The term Naïve Bayes refers to the fact that the probability can be derived using Bayes theorem and that it incorporates strong independences, hence are naïve.

Naïve Bayesian Classification works as follows:

- Each data sample is represented by an n-dimensional feature vector X = ($x_1$, x2….,xn), depicting *n* measurements made on the samples from n attributes, respectively $A_1, A_2 \ldots A_n$.
- Suppose that there are m classes, C1, C2,…..Cn. Given an unknown data sample, X (having no class label), the classifier will predict that X belongs to the class having the highest posterior probability conditioned on X (PX|H) i.e. Naïve Bayes Classifier assigns unknown sample X to class $C_i$ iff

$$P\ (C_i|X) > P\ (C_j|X) \text{ for } 1 \leq j \leq m, j \neq i$$

$$P\ (C_i|X) = [P\ (X|C_i) \,.\, P\ (C_i)] / P\ (X)$$

- As P (X) is constant for all classes, only P ($X|C_i$). P (Ci) needs to be maximized. If the class prior probabilities are not known, then it is commonly assumed that the classes are equally likely $P(C_1) = P(C_2) = \ldots P(C_m)$ and would maximize P ($X|C_i$). Otherwise, P ($X|C_i$). P ($C_i$) is usually maximized.

  Class prior probability can be estimated by:

$$P\ (C_i) = S_i / S$$

  where $S_i$ is the number of training samples of class $C_i$

  S is the total number of training samples.

- Given datasets with many attributes, to reduce the computation in evaluating P ($X|C_i$)

  The naïve assumption of class conditional independence is made.

$$P\ (X|C_i) = \prod^n_{k=1} P\ (x_k|C_i).$$

  The probabilities P ($x_1|C_i$), P ($x_2|C_i$), ….P ($x_n|C_i$) can be estimated from the training samples.

- In order to classify an unknown sample X, P ($X|C_i$) . P ($C_i$) is evaluated for each sample $C_i$. Sample X is then assigned to the class $C_i$ iff

$$P\ (X|C_i) \,.\, P\ (C_i) > P\ (X|C_j) \,.\, P\ (C_j) \text{ for } 1 \leq j \leq m, j \neq i$$

*2.2.2.5 Extraction of new keyphrases*

To select keyphrases from a new document, Kea determines candidate phrases and feature values and applies the model built during training.

When the Naïve Bayes model is used on candidate phrase with feature values t (TF x IDF) and d (distance), two quantities are computed:

$$P[yes] = \frac{Y}{Y+N} P_{TFxIDF}[t \mid yes]\, P_{distance}[d \mid yes]$$

And a similar expression for P [*no*], where Y is the number of positive instances in the training files (author identified keyphrases) and N is the number of negative instances (candidate phrases that are not keyphrases).

The overall probability that the candidate phrase is a keyphrase can be calculated by

$$p = P[yes] / (P[yes] + P[no])$$

Candidate phrases are ranked according to this value and two post-process steps are carried on:

- TF x IDF is used as a tiebreaker if two phrases have equal probability.
- Remove from the list any phrase that is subphrase of a higher-ranking phrase.

Kea [25], is based on the naive Bayes machine learning technique. The basic model involves 2 attributes: distance and TF*IDF (Term Frequency times Inverse Document Frequency). Kea algorithm considers keyphrase extraction as a classification problem and uses Naïve Bayes algorithm. Experiments show that the computational complexity of Kea's training is much lower and thus it is possible to train Kea for domain specific tasks. In this thesis we have used Kea [25] algorithm for automatic keyphrase extraction.

### 2.2.3 The LAKE System for Keyphrase Extraction

There exist many other algorithms to extract keyphrases like LAKE (Learning Algorithm for Keyphrase Extraction), which makes use of linguistic processing of documents to extract candidate phrases from a given document [5] and then it uses a machine learning approach to select the significant keyphrases for that document. Extraction of keyphrases using LAKE system involves initial preprocessing of the given document as follows:

- Part of speech tagging of the given input document
- Recognition of sequences that are single lexical units according to their presence in WordNet.
- Named entity recognition i.e. identification and the categorization of entity names (person, location, organization etc), temporal expressions (dates, time), numerical expressions (measures, percentages) in written texts.

#### *2.2.3.1 Candidate phrase extraction*

Candidate phrases are formed based on the output of part of speech tagger. The candidate phrases should match one of the pre-defined syntactic patterns (for instance, *Named Entity, noun, adjective+noun, noun+verb+adjective+noun etc.).* Once the list of candidate phrases is extracted, scoring the phrases to identify the most significant phrases is performed. The features for candidate phrases TF x IDF and the first occurrence for each candidate phrase are calculated. Naïve Bayes Classifier is used to classify the candidate phrases as a keyphrase or a non-keyphrase. From a document collection all the nouns and verbs are extracted. Each of them is marked as a positive example or a negative example depending on their match with author assigned keyphrases. This classifier is run on a new document to find out keyphrases.

# Chapter 3
# Summarization

With the coming of the information revolution, electronic documents are becoming a principle media of business and academic information. Thousands and thousands of electronic documents are produced and made available on the internet each day. In order to fully utilize these on-line documents effectively, it is crucial to be able to extract the gist of these documents. In most cases, summaries are written by human. But nowadays, enormous amount of information is available on-line and its impossible for any human to summarize all the available textual information. The need to access the essential content of documents accurately to satisfy users' demands, calls for the development of computer programs which are able to produce text summaries.

Research and development in the area of automatic text summarization has been growing in importance with the rapid growth of the Web and on-line information services. Summarization is the art of abstracting key content from one or more information sources. People keep abreast of world affairs by listening to news bites. They even go to movies largely on the basis of reviews. With summaries, they can make effective decisions in less time.

## 3.1 Why do we need Summaries?

The simplest answer could be, in order to gain access to and control the flood of information, everyone needs to know in brief what is worth reading and what is useful for a particular purpose. Nobody wants to waste time reading what is useless. By giving an overview or outline of content, summaries save readers' time. Some contexts in which summarization is important are:

- *Abstracts*

Abstracts are a vital component of communication of research, saving the reading time of individual researchers and improving the control of information.

➢ *Review articles*

Typically an article reviewing progress in a specific research field covers a wide range of documents. In some cases, only the bare contents of texts are mentioned (indicative); in others, more substance is reported (informative). But most importantly, the review article weighs up the current status and indicates the important contributions (evaluative and selective).

➢ *Encyclopedias*

Encyclopedia articles review the state of the art in a more global fashion. They are evaluative (and almost necessarily selective) summaries of 'what is known' about a particular topic (informative). They represent 'starting points' for readers, and hence frequently refer to other encyclopedia articles or 'further reading' (in this respect they are indicative).

➢ *Journalism*

Summaries are the stock in trade of most journalists. Many newspaper reports are extracts from other texts (e.g. reports of debates and official documents). Most journalist summaries are selective (often evaluative).

➢ *Market surveys and reports*

These basic information sources for business people are intrinsically compilations of summaries of documentation produced by companies and of evaluations of products.

## 3.2 Approaches for Text Summarization

There are two fundamental approaches to automatic text summarization: Extracted Summaries and Abstracted Summaries. Both result in the compression of text, but one is relatively shallow, while the other is deep and complex.

### *3.2.1 Extracted Summary*

On the least-complex end is summarization through *text extraction*, the creation of summaries using terms, phrases and sentences pulled directly from the source text using statistical analysis at a surface level. Occurrences of words or sentences are counted and analyzed according to their frequency and where they appear and reappear in the source text. Karen Spark Jones [18], views text extraction as "what you see is what you get," because parts of source text are extracted directly. It is thus an "open" approach that determines "importance" mechanically. The extracted text may well be incoherent if pronouns, synonyms or other ambiguous terms aren't sufficiently resolved.

*The following example shows the extracted summary of the text given in Fig 4:*

```
Fourscore and seven years ago our fathers brought forth upon this continent a new nation, conceived in liberty, and dedicated to the proposition that all men are created equal. Now we are engaged in a great civil war, testing whether that nation, or any nation so conceived and so dedicated, can long endure. The brave men, living and dead who struggled here, have consecrated it far above our power to add or detract.
```

TH-12527

Fig. 4: An example of source text

*Extract obtained for the text given in text:*

```
Now we are engaged in a great civil war, testing whether that nation or any nation so conceived and so dedicated, can long endure
```

### *3.2.2 Abstracted Summary*

The other, more complex and "knowledge-rich" endpoint is summarization through *abstracting.* The aim here is to turn a computer-generated analysis and synthesis of the source material into a completely new, shorter text that is still cohesive and intelligible. In other words, it reads as though a human had written it, and at the same time fulfills the specific information need of the user. This process is sometimes known as *machine understanding,* a multidisciplinary effort involving information retrieval, linguistics and artificial intelligence.

In simplest terms, automatic abstracting is *fact extraction.* Fact extraction, on the other hand, is "closed" as the process requires pre-established, domain-determined parameters for machine processing. Spark Jones [18] views text abstraction as, "What you see is what you know," because the system assembles facts based on these requirements and may altogether ignore what the authors of the source material felt was important.

The direct development of ontologies, controlled vocabularies, thesauri and other agents of understanding are what make this automated abstracting so challenging and will keep researchers engaged for many years to come. But the need to deal with information overload is immediate and that is why the text extraction model is being so actively pursued. For one reason, because it does not require creation of an entirely new text, expectations for it are quite a bit lower.

*Abstract generated for the text given in Fig.4*

```
This speech by Abraham Lincoln commemorates
soldiers who laid down their lives in the Battle
of Gettysburg. It offers an eloquent reminder to
the troops that it is the future of freedom in
America that they are fighting for.
```

## 3.3 Various Approaches to Generate Summaries by Extraction

The main focus of this thesis is on summarization by extraction. Most of the work on summarization carried out in recent times is geared towards the extraction of significant text fragments from a document. This extraction process can be classified into two broad categories:

- **domain dependent approaches** where a priori knowledge of the discourse domain and text structure (e.g. weather, financial, medical) is exploited to achieve high quality summaries.

- **domain independent approaches** where a statistical (e.g. vector space indexing models) as well as linguistic techniques (e.g. lexical cohesion) are employed to identify key passages and sentences of the document.

### *3.3.1 Domain Dependent Approaches*

Several domain dependent approaches to summarization use Information Extraction techniques in order to identify the most important information within a document. Work in this area also includes techniques for Report Generation and Event Summarization from specialized databases.

### *3.3.2 Domain Independent Approaches*

Most domain-independent approaches often use statistical techniques in combination with shallow language technologies to extract salient document fragments. The statistical techniques used are similar to those employed in Information Retrieval and include vector space models, term frequency and inverted document frequency.

Few Statistical techniques used for text summarization:

- Length of sentence
- Indicators like "In conclusion...", "We found..."
- Structure of paragraphs: Beginnings and ends are important
- Keywords: Frequency of content words

## 3.4 Summarization Techniques Studied

Many techniques have already been developed for automatic text summarization. These approaches have used a set of indicators such as cue phrases, term frequency, and sentence position to choose sentences to form into a summary. Few approaches used for summarization are discussed below:

### 3.4.1 Trainable Summarizer

This system developed by Kupiec, Pederson [16], focuses on document extracts, a particular type of computed document summary. The system extracts the sentences based on the following discrete feature set for the given text:

Sentence Length Cut-off Feature, Fixed-Phrase Feature, Paragraph Feature, Thematic Word Feature, Uppercase Word Feature.

For each sentence *s*, the probability of *s* being included in the summary is calculated based on the k given features $F_j$ ; $_{j = 1.......k.}$ , which can be expressed using the Bayes' rule as follows:

$$P ( s \in S \mid F_1,F_2,....F_k) = P (F_1,F_2,....F_k \mid s \in S) P ( s \in S) / P (F_1,F_2,....F_k)$$

Assuming statistical independence of the features

$$P ( s \in S \mid F_1,F_2,....F_k) = \prod^k_{j=1} P (F_j \mid s \in S) P ( s \in S) / \prod^k_{j=1} P (F_j)$$

P (s $\in$ S) is a constant and P ($F_j$ | s $\in$ S) and P ($F_j$) can be estimated directly from the training set by counting occurrences. This yields a simple Bayesian classification function that assigns for each sentence *s* a score which can be used to select sentences for inclusion in the summary.

Training procedure include to obtain the corresponding match between the manual summary sentences and sentences in the original document. Sentences from the original documents can be matched to those in manual summaries in several ways. A *direct sentence* match occurs when a manual summary sentence could either be extracted directly from the original or with minor modifications, preserving the content. When it is obvious that two or more sentences were used from the original to make a summary sentence, a *direct join* occurs. If it is suspected that the expert constructed a summary sentence from a general reading, the summary sentence is marked as *unmatchable.*

When the overlap occurs between summary sentence and the original but the content of original is not preserved in summary sentence or when the summary sentence includes a sentence from the original documents but also contains other information that is not covered in direct join are marked as *incomplete.*

A sentence produced by the summarizer is defined as correct if:

- It has a direct match, and is present in the manual summary.
- It is in the manual summary as part of a direct join, and all other members of the join have also been produced.

### 3.4.2 Lexical Chains for Text Summarization

The summarization system developed by Barzilay and Elhadad [1] describes lexical chain identification in a given text for extracting sentences to form summaries. A lexical chain is a set of semantically related words in a text. To identify the relationship between two words WordNet lexical database can be used. WordNet can be considered as a dictionary containing definitions of each word. In WordNet each word and its different senses are stored. The relations between words are stored in WordNet database. Lexical chains cannot be obtained with a surface analysis of the text. Thus the text is passed through structural analysis and chains are constructed by discovering the semantic relation between words.

A procedure for constructing lexical chains follows three steps:

1. Select a set of candidate words.
2. For each candidate word, find an appropriate chain relying on a relatedness criterion among the members of the chains.
3. If it is found, insert the word in the chain and update it accordingly.

In the preprocessing step all the words that appear as a noun entry in WordNet are chosen. Three kinds of relations are defined: extra-strong (between a word and its repetition), strong (between two words connected by a WordNet relation) and medium-strong when the link between the synsets of the words is longer than one.

To obtain the summary of any given text it is necessary to identify the strongest chains among all those produced by the algorithm. But there is no formal method to evaluate the chain strength. So chain strengths are obtained by empirical methods.

Once strongest chains are selected, the next step of the summarization algorithm is to extract full sentences from the original text.

- For each chain in the summary representation choose the sentence that contains the first appearance of a chain member in the text.
- For each chain in the summary representation, choose the sentences that contains the first appearance of a representative chain member in the text.
- For each chain, find the text unit where the chain is highly concentrated. Extract the sentence with the first chain appearance in this central unit. Concentration is computed as the number of chain members occurrences in a segment divided by the number of nouns in the segment. A chain has high concentration if its concentration is the maximum of all chains.

### 3.4.3 Extracting Sentence Segments for Text Summarization

The system proposed by Wesley Chuang, Jihoon Yang [4] extracts sentence segments to generate a summary. The working of the system in brief is as follows: first the sentences are broken into segments by special cue-markers. Each segment is represented by a set of predefined features. Then a supervised learning algorithm is used to train the summarizer to extract important sentences, based on feature vector.
A sentence is segmented by a cue-phrase. The basic idea behind this is to separate out sentence segments that possibly convey independent meaning. The purpose of sentence segmentation is to use sentence segments as a basic unit for summarization.

The sentence segments are represented by structured set of features like rhetorical relations. In a complex sentence, with two clauses, the main segment is called the *nucleus*, and its subordinate segment is called a *satellite* and is connected to the main segment by some kind of rhetorical relation. Generally, when a rhetorical relation occurs, the nucleus is more important and has more chance to be included in the summary. A rhetorical relation *r (name, satellite, nucleus)* shows that there exists a relation of type *name* between the *satellite* and the *nucleus* segments.

A total of 23 features are collected to generate a feature vector like:

paragraph number, offset in the paragraph., number of bonus words, number of title words, average term frequency etc.

The goal is to select few segments as a summary of original text. With the feature vectors generated in the previous steps, a machine learning algorithm can be applied to train the summarizer. Naïve bayes Classifier is used in this system, to decide whether a particular sentence segment will be included in the summary or not, as it is easy and powerful.

### 3.4.4 Cut-Paste based Text Summarization

Cut and Paste method proposed by Hongyon Jing [13] is a summarization technique, which initially extracts key sentences from a given article using Lexical Chains method as described above. The words in the sentence are linked with other words in the article through the lexical relations encoded in WordNet. An importance score is computed for each word in a sentence based on the number of lexical links it has with other words taking into consideration the type of links and the direction of links. Once each word is assigned a score then sentence score for each sentence is computed by adding up the scores for each word. The sentences with higher scores are considered important. This system also uses many other methods to extract important sentences like sentence positions, cue phrases, tf*idf scores. Once the important sentences are identified the summarizer uses reduction to remove inessential phrases and combination to merge resulting phrases together as coherent sentences.

# Chapter 4
# Summarization using Keyphrase Extraction

As the amount of on-line information increases, systems that can automatically summarize one or more documents become increasingly desirable. Recent research has looked into types of summaries, methods to create them, and methods to evaluate them. Most work today still focuses on extraction of sentences from the original document to form a summary. Recent extraction approaches use more complicated techniques for deciding which sentences to extract, these techniques often rely on machine learning to identify important features.

## 4.1 Summarization using Keyphrases

The present thesis focuses on summarization of text documents using automatically extracted keyphrases. As discussed in the previous chapters, the main goal of keyphrases is to provide the user with the overview of document's contents. Keyphrases are useful for a variety of purposes, including summarizing, indexing, labeling, categorization, clustering, highlighting, browsing and searching. Our key interest is in the area of summarization. In order to generate a summary, we have to identify the most important pieces of information from the document, omitting irrelevant information and minimizing details, and assemble them into a compact coherent report. These important pieces of information are the keyphrases for the document. Using these keyphrases the most important sentences from the document can be selected to give the summary for that document.

As discussed in the previous chapter, few statistical techniques used for text summarization are

Length of sentence, Indicators like "In conclusion", "We found", Structure of paragraphs Beginnings and ends are important, Keywords: Frequency of content words

In this thesis we will be using keywords for text summarization. Not much work has been done on *summarization using keyphrase extraction* as it is the most simple technique and the researchers are interested in solving much complex problems involved in text processing. So as a part of M.Tech thesis we have decided to develop summarization method using automatically extracted keyphrases.

In the present work we have planned to use domain independent statistical approach for text summarization. The summarization method involves extracting the important keyphrases from a given document using KEA [7, 25] keyphrase extraction algorithm and then using these extracted keyphrases to generate sentences consisting of topmost keyphrases. Sentences are extracted by a direct match between the keyphrases extracted using Kea and the ones present in the sentence [19]. Another technique, Mutual reinforcement principle [14] is also employed to select the important sentences from the text using the keyphrases to form the summary.
Once the summary is obtained, as a step towards producing cohesive summary, a sentence ordering scheme [10, 20] is applied to the sentences in the extracted summary.

**4.2 Automatic Extraction of Keyphrases**

As discussed earlier keyphrases give a very short summary of a document, which is intended to make it easy for readers to quickly determine if the document is relevant to their information needs. Keyphrase extraction is a classification task: a document can be seen as a set of phrases, and a keyphrase extraction algorithm should correctly classify a phrase as a keyphrase or a non-keyphrase. Machine learning techniques can automate this task if they are provided with a set of training data: examples of both keyphrases and non-keyphrases. The data are used to teach the algorithm how to distinguish keyphrases from non-keyphrases. The resulting algorithm can then be applied to new documents for keyphrase extraction.

Kea [25], keyphrase extraction algorithm has been used for the automatic extraction of keyphrases from the given text. The details of the algorithm are discussed in chapter 2.

## 4.3 Document Summarization

The output of Kea will be a set of keyphrases and these keyphrases are used to extract summary from the given document using direct extraction of important sentences [19] and Mutual Reinforcement Principle [14]. The details of how to extract sentences using the keyphases are as follows:

### *4.3.1 Direct Extraction of Important Sentences*

In order to generate a summary for a given document using automatically extracted keyphrases, we need to identify the important sentences in the given document. The importance of a sentence can be judged by the presence of keyphrases in that sentence. As discussed above Kea assigns each keyphrase a score to mark its importance in the text. Depending on these scores of keyphrases, the scores of the sentences can be determined. Sentences in the document are then ranked in order of importance. The highest-ranking sentences will be selected as the summary for the given text documents.

### *4.3.2 Mutual Reinforcement Principle to Extract Important Sentences*

In this method, keyphrases and the sentences containing them are modeled as weighted undirected and bipartite graphs. The process of summarization is achieved as follows:

For each document, two sets of objects are generated:

- ➢ set of terms T= $\{t_1,t_2.....t_n\}$, the keyphrases extracted above will be the set of terms in T and
- ➢ set of sentences S=$\{s_1,s_2....s_m\}$.

A bipartite graph is built from T to S in the following way:

If the term $t_i$ appears in the sentence $s_j$, then create an edge between $t_i$ and $s_j$.

Weights for the edges are the number of times a term $t_i$ appears in sentence $s_j$. The weighted bipartite graph is denoted by G (T, S, W) where W is the weight matrix containing all the pairwise edge weights.

The salience score is computed for each term $t_i$ as u($t_i$) and each sentence $s_j$ as v($s_j$). The reinforcement principle states as follows:

*A term should have a high saliency score if it appears in many sentences with high saliency scores while a sentence should have a high saliency score if it contains many terms with high saliency scores* [14].

The principle dictates that the saliency score of a term is determined by the saliency scores of the sentences it appears in, and the saliency score of a sentence is determine by the saliency scores of the terms it contains.

While computing the term scores, the summation is over all the sentences that contain the term and while computing the sentence scores, the summation is over all the terms that appear in the sentence.

The saliency scores for terms and sentences are collected into two vectors

$$u = 1/\sigma\, Wv \qquad\qquad v = 1/\sigma\, Wu$$

where $\sigma$ is the proportionality constant

The corresponding component values of u and v give the term and sentence saliency scores. Rank sentences in decreasing order of their saliency scores, and select the top *s* sentences to add to the summary.

The above-specified technique is used to select important sentences from a given document. The keyphrases are the set of terms *T* and all the sentences in the document will be in the set *S*. A bipartite graph is built from set *T* to set *S*. Weights are assigned to the edges of the bipartite graph from *T* to *S* depending on number of times particular term occurs in a particular sentence. Each term's importance is determined by the number of sentences in which it is present and that count will be the score for that term. These scores are stored for later use. Each sentences' importance is determined by the number of terms it contains i.e. the score of a sentence is determined as the sum of all the scores of the keyphrases present in the sentence. Hence the scores for the sentences are also obtained. The sentences are then ranked in the decreasing order of their scores and top ranking sentences are selected to give the summary of the given document.

One issue that has received little attention is how to organize selected information so that the output summary is coherent. Once all the relevant pieces of information have been selected across the input document, the summarizer has to decide which order to present them to the user. Inorder to present a cohesive summary as output, a sentence ordering scheme has been applied to the sentences in the extracted summary which is based on the internal links between keyphrases of the extracted text.

## 4.4 Generating Cohesive Summaries

### *4.4.1 Cohesion*

A text or discourse is not just a set of sentences, each on some random topic. Rather, the sentences and phrases of any sensible text will each tend to be about the same things i.e. the text will have a quality of relatedness. This is the property of *cohesion*, the sentences "stick together" to function as a whole. Cohesion is achieved through back-reference, conjunction and semantic word relations. Cohesion is not a guarantee of unity in text but rather a device or technique for creating it.

Cohesion is defined as the way certain words or grammatical features of a sentence can connect it to its predecessors (and successors) in a text. The simplest definition of cohesion is that it "refers to relations of meaning that exist within the text and that define it as a text". Cohesion connects a string of sentences to form a text rather than a series of unrelated statements. Halliday [11] defined cohesion as "the set of possibilities that exist in the language for making text hang together". Studies of cohesion show that cohesion makes a substantial contribution to readability and this is the reason why study of cohesion in text is of interest. Cohesion occurs when the interpretation of some element in the discourse is dependent on that of another.

For example, in the sentences "*John ate the apple. He thought it was delicious.*" the word *it* in the second sentence refers back to *apple* in the first sentence, and the word *he* refers back to *John*. Cohesion holds segments of a text together, making it a semantic edifice, just as mortar does bricks or stones in a building. The importance of cohesion lies in the continuity it expresses between one part of the text and another. This continuity is necessary for the interpretation of text.

### *4.4.2 Cohesive Summary Generation*

A summary of certain fixed number of sentences is obtained from the above techniques. The extracted sentences are picked up randomly from the input text based on some measure like keyphrase score etc. Since the extracted sentences are disconnected in the original article, when they are put together, the resulting summary can be inconcise, incoherent, and sometimes insensitive.

In order to create a cohesive summary a sentence ordering scheme is applied to the extracted summary using the internal links between keyphrases [10, 20]. Some keyphrases are found in more than one sentence. Hence each sentence has some forward or backward mentions. Extracting those sentences that have a good combination of forward and backward mentions can make up a good summary.

The summary obtained consists of sentences having keyphrases. Hence each sentence will have some forward and backward mentions. We need to select sentences, which have a good combination of forward and backward mentions. This requires computation of mean for a sentence, which is the average of forward and backward mentions of the sentence normalized over the number of keyphrases present in the sentence. Deviation is computed as the difference of normalized mean with the forward mentions or backward mentions. The sentences are sorted in the increasing order of their means and decreasing order of their deviations. The resultant will be a sensitive, well-ordered cohesive summary. We can extract the top sentences in the list to give a brief, sensitive summary instead of going through the complete extracted summary. This method gave a satisfactory output. For a brief summary topmost sentences can be extracted.

# Chapter 5
# Implementation and Results

The algorithms Kea (for keyphrase extraction), direct selection of sentences and mutual reinforcement principle for summarization are implemented in C language. Experimental results using different examples and a discussion on the results obtained are presented in this chapter.

The input to the system is a plain text file. Keyphrases are extracted from the given text using Kea, keyphrase extraction algorithm, the details of which are mentioned in chapter 2. These keyphrases will be used for summarization purpose as specified below.

## 5.1 Method 1(Direct sentence selection)

Using the keyphrases obtained from Kea, summary is generated by direct extraction of sentences containing the top scored keyphrases. Sentence score is calculated by summing up of all the scores of keyphrases present in it. Scored sentences are sorted in the decreasing order of their scores and top scoring sentences are selected to give the summary. We select the top six sentences as the summary of the given text.

## 5.2 Method 2(Mutual reinforcement principle)

The second method used to generate summary using already extracted keyphrases is by using Mutual Reinforcement Principle. This principle states that "*a term or keyphrase has a high score if it appears in many sentences while a sentence has high score if it contains many terms or keyphrases in it with high scores.*" In this method a weighted bipartite is constructed between the keyphrases and sentences of the given input text. Depending on the presence of a keyphrase in a sentence edges are built, i.e. if a keyphrase is present in the sentence then there exists an edge between the term and the sentence. The weight for the edge is the number of times term appears in the sentence. Depending upon the number of sentences in which a particular keyphrase is present the score for that keyphrase is calculated. In this way each keyphrase will be assigned a score. Then sentence scores are computed by summing up of all the keyphrase's scores present in a particular sentence. Once each sentence is assigned a score they are arranged in the decreasing order of their scores. Top scoring sentences are extracted to give the summary of the given text.

## 5.3 Sentence Ordering Scheme

The summary obtained will be a random selection of sentences from the given text depending on the scores of keyphrases present in them. Inorder to make the extracted text sensitive and cohesive a sentence ordering scheme [20] is applied to the sentences of extracted summary. This scheme uses the internal links between the keyphrases of the extracted sentences. Some keyphrases can be present in more than one sentence. So we need to select sentences with a good combination of forward and backward mentions to get a meaningful summary. The forward and backward mentions for each sentence are computed and their mean and deviation are obtained. Sentences are sorted in the increasing order of their means and decreasing order of their deviations. The sentences arranged in the sorted order gives a well-ordered and cohesive summary. Even the top few sentences can be selected to give the brief summary of the text.

## 5.4 Results

The results obtained from the above specified summarization methods are as follows:

**Example 1:**

The contents of the input file:

The task of automatic Document **Summarization** is to take an information source, extract content from it, and present the most important content to the user in a condensed form and in a manner sensitive to the user's or application's needs. This area is highly interdisciplinary and related with natural language processing, artificial intelligence, information retrieval, information extraction, statistics and cognitive psychology. Over the last few years, much of the research has been concentrated in finding effective algorithms and building up efficient systems, such as SMART, SUMMARIST, MEAD, NEATS, WebInEssence and the Columbia Multi-Document Summarizer which provided precious experience for future research. As Automatic Document **Summarization** is closely related with cognitive psychology as well, much attention and study have been paid in psychological models for text understanding and representations, for example, the Construction-Integration model and the **Event-Indexing** model.

To summarize a document or documents, a reader has to understand the document and integrate information and make connections across sentences to form a coherent discourse representation. **Syntax** is there in order to allow the construction of formal structures by means of which complex meanings can be expressed. Considering the importance of sentences, **syntax** and phrases, and being inspired from the **Event-Indexing** model, we designed and developed a new generic algorithm for automatic document **summarization** based on the analysis of human cognition and intelligence. We hereby introduce this algorithm, named **Event-Indexing** and **Summarization** algorithm, which applies the indices from the **Event-Indexing** model for sentence indexing, using syntactic parsing for sentence analysis, and WordNet for phrase level analysis and processing. Zwaan, Langston and Graesser have explained and tested their **Event-Indexing** model, which means that readers monitor five aspects or indices of the evolving situation model when they read stories: Protagonists, Temporality, Causality, Spatiality, Intention. To make this model applicable to computing, we redefined the concepts of 'event', 'Protagonist', 'Temporality', 'Spatiality', 'Causality' and 'Intention' as below. 'Event' is a cognitive psychological concept, and can be either a story or a sentence in microstructure. Zwaan and Radvansky treated each sentence as an event, in their paper. So we also render the equal concepts of 'event' and 'sentence'. In this context, 'Protagonist' can be considered as the subject or noun phrase that plays the role of subject of each sentence. 'Temporality' is the temporal information contained in each sentence. 'Spatiality' is the space or location information in each sentence. 'Causality is the casual relationship of a sentence to previous sentences or contained in one sentence. 'Intention' is the relationship between a subject's goal and sentences in the document. The EIS algorithm includes a syntactic parser, the Link Grammar Parser for parsing sentences syntactically, index extraction and indexing modules, which applied the WordNet, a lexical database for the English Language, with the guidance of the new definitions of the five indices from the **event-indexing** model, and other related modules. The Document Re-construction module transforms text files of HTML format into plain text format, by using the modified James Clark's XML parser. The transformation of document to sentences module uses some regular expressions to split the text section extracted from documents into sentences. As it is known that some abbreviations cause incorrect segmentation, a collection of such abbreviations has been made to reduce segmentation errors. This module breaks the original structure of each document but records the original information of the position of each sentence in the document. According to the size of each retained cluster and the total size of all retained clusters, we configure the percentage that each retained cluster should occupy in the final summary. If an extracted summary exceeds the required summary size say 100 words, the summary will be returned to the module of cluster reduction for further word reduction. Once the summary is qualified at the required size, it will become an output of a formal result of **summarization**. The multiple documents **summarization** results of the first draft if this algorithm are based on the DUC 2003 evaluations which took place in Feb 2003. Multiple document **summarization** results of 18 systems including 2 guideline systems, were evaluated by DUC2003. After some improvements of the EIS algorithm in pronoun resolution and causality indexing were made, we re-evaluated multi-document **summarization** with the document collection, DUC 2001, and its 100-word model summaries for 30 groups; each group contains 7 t o16 documents.

The keyphrases obtained are: **summarization, event-indexing, syntax**

Summary generated by direct extraction of sentences from the text:

1.We hereby introduce this algorithm, named **Event-Indexing** and **Summarization** algorithm, which applies the indices from the **Event-Indexing** model for sentence indexing, using syntactic parsing for sentence analysis, and WordNet for phrase level analysis and processing.
2.Considering the importance of sentences, **syntax** and phrases, and being inspired from the **Event-Indexing** model, we designed and developed a new generic algorithm for automatic document **summarization** based on the analysis of human cognition and intelligence.

3.As Automatic Document **Summarization** is closely related with cognitive psychology as well, much attention and study have been paid in psychological models for text understanding and representations, for example, the Construction-Integration model and the **Event-Indexing** model.
4.Zwaan, Langston and Graesser have explained and tested their **Event-Indexing** model, which means that readers monitor five aspects or indices of the evolving situation model when they read stories: Protagonists, Temporality, Causality, Spatiality, Intention
5.The EIS algorithm includes a syntactic parser, the Link Grammar Parser for parsing sentences syntactically, index extraction and indexing modules, which applied the WordNet, a lexical database for the English Language, with the guidance of the new definitions of the five indices from the **event-indexing** model, and other related modules.
6.Once the summary is qualified at the required size, it will become an output of a formal result of **summarization.**

Summary generated by Mutual Reinforcement Principle

1.We hereby introduce this algorithm, named **Event-Indexing** and **Summarization** algorithm, which applies the indices from the **Event-Indexing** model for sentence indexing, using syntactic parsing for sentence analysis, and WordNet for phrase level analysis and processing.
2.Considering the importance of sentences, **syntax** and phrases, and being inspired from the **Event-Indexing** model, we designed and developed a new generic algorithm for automatic document **summarization** based on the analysis of human cognition and intelligence.
3.As Automatic Document **Summarization** is closely related with cognitive psychology as well, much attention and study have been paid in psychological models for text understanding and representations, for example, the Construction-Integration model and the **Event-Indexing** model.
4.The task of automatic Document **Summarization** is to take an information source, extract content from it, and present the most important content to the user in a condensed form and in a manner sensitive to the user's or application's needs.
5.Once the summary is qualified at the required size, it will become an output of a formal result of **summarization.**
6.The multiple documents **summarization** results of the first draft if this algorithm are based on the DUC 2003 evaluations which took place in Feb 2003.

Computing forward and backward mentions for sentences in the resultant summary:

| Sentence Number | Backward count (B) | Forward Count(F) | No. of keywords in sentence | Mean M= (B/N+F/N)/2 | Deviation D= abs(M-F) |
|---|---|---|---|---|---|
| 2. | 2 | 4 | 3 | 1 | 3 |
| 1. | 0 | 5 | 2 | 1.25 | 3.75 |
| 3. | 4 | 3 | 2 | 1.75 | 1.25 |
| 6. | 5 | 0 | 1 | 2.5 | 2.5 |
| 5. | 4 | 1 | 1 | 2.5 | 1.5 |
| 4. | 3 | 2 | 1 | 2.5 | 0.5 |

The resultant ordering of the sentences obtained is 2,1,3,6,5,4.

**The output Sequence after computing the forward and backward mentions:**

2.Considering the importance of sentences, **syntax** and phrases, and being inspired from the **Event-Indexing** model, we designed and developed a new generic algorithm for automatic document **summarization** based on the analysis of human cognition and intelligence.
1.We hereby introduce this algorithm, named **Event-Indexing** and **Summarization** algorithm, which applies the indices from the **Event-Indexing** model for sentence indexing, using syntactic parsing for sentence analysis, and WordNet for phrase level analysis and processing.
3.As Automatic Document **Summarization** is closely related with cognitive psychology as well, much attention and study have been paid in psychological models for text understanding and representations, for example, the Construction-Integration model and the **Event-Indexing** model.
6.The multiple documents **summarization** results of the first draft if this algorithm are based on the DUC 2003 evaluations which took place in Feb 2003.
5.Once the summary is qualified at the required size, it will become an output of a formal result of **summarization**.
4.The task of automatic Document **Summarization** is to take an information source, extract content from it, and present the most important content to the user in a condensed form and in a manner sensitive to the user's or application's needs.

**Example 2:**

The contents of the input file:

As our first participation in DUC, we developed LAKE, a system that exploits the role of **Keyphrase Extraction** as a useful approximation to **summarization**, evaluating its performance in task. Our decision to participate was mainly motivated by the fact that some features of task, the length limit of the output summaries and the fact that summaries could be returned as lists of disjointed items, seemed to fit well in a KE approach. Keywords or keyphrases, provide semantic metadata that characterize documents, producing an overview of the subject matter and contents of a document. Keyword extraction is a relevant technique for a number of text-mining related tasks, including document retrieval, Web page retrieval, document clustering and **summarization**, Human and Machine Readable Indexing and Interactive Query Refinement. There are two major tasks exploiting keyphrases: keyphrase assignment and **keyphrase extraction**. In a keyphrase assignment task there is a predefined list of keyphrases (i.e. a controlled vocabulary or controlled index terms). These keyphrases are treated as classes, and techniques from text categorization are used to learn models for assigning a class to a given document. A document is converted to a vector of features and **machine learning** techniques are used to induce a mapping from the feature space to the set of keyphrases (i.e. labels). The features are based on the presence or absence of various words or phrases in the input documents. Usually a document may belong to different classes. In **keyphrase extraction** (KE), keyphrases are selected from the body of the input document, without a predefined list. When authors assign keyphrases without a controlled vocabulary (free text keywords or free index terms), typically about 70% to 80% of their documents [10]. This suggests the possibility of using author-assigned free-text keyphrases to train a KE system. In this approach, a document is treated as a set of candidate phrases and the task is to classify each candidate phrase as either a keyphrase or non-keyphrase[10, 4]. A feature vector is calculated for each candidate phrase and **machine learning** techniques are used to learn a model, which classifies each candidate phrase as a keyphrase or a non-keyphrase. The paper is organised as follows. In section 2 we report on the general architecture of our system, which combines a **machine learning** approach with a **linguistic processing** of the document. Section 3 shows the results obtained by the system and discusses the evaluation carried out with ROUGE. We conclude suggesting possible future improvements. In this section we describe LAKE (Learning Algorithm for Keyphrase Extraction), a **keyphrase extraction** approach developed at ITC-irst based on a supervised learning approach that makes use of a **linguistic processing** of the documents. The system works in two phases (see figure 1): first it considers a number of well-motivated candidate keyphrases from a given document: then it uses a **machine learning** framework to select significant keyphrases for that document. More in detail, candidate phrases consist of words or sequences of words that match a set of previously manually defined linguistic patterns. Then a supervised learning algorithm is used to score the head of each phrase, according to features that signal the relevance of the head in the whole document collection. The motivation for considering the heads of the candidate phrases instead of the phrase itself is that phrases do not appear frequently enough in the collection. Finally, the score of the head is assigned to the whole candidate phrase and the best scores phrases that fills the 75 bytes required by the task are given as output. Both the linguistic patterns and features used in the classification phase have been defined considering the duc-2003 material. The document collection provided by DUC was preprocessed according to the following three steps: part of speech tagging, multiword recognition, named entity recognition. We use the Treetagger POS tagger developed at University of Stuttgart. The tagged text is given to a module that recognizes multiword expressions. Sequences of words that are considered as single lexical units are detected in the input document according to their presence in WordNet. As for Names Entities recognition we used NERD, a multilingual Named Entity Recognizer for Italian and English. The system has been designed for the identification and the categorization of entity names, temporal expressions and certain type of numerical expressions in written texts. The selection of relevant phrases has been strongly task-oriented. Within the framework of the DUC **summarization** task, the concept of relevance was heuristically defined: we considered significant the syntactic patterns that described either a precise or well-defined entity, or concise event. In the former case we focussed on uni-grams and bi-grams, while in the latter we considered longer sequences of parts of speech, often containing verbal forms. Despite the important role heuristics play in our approach, the choice of relevant patterns was also linguistically motivated: sequences like noun + adjective, that are not allowed in English were taken into consideration. We manually selected a restricted number of PoS sequences that could have been significant in order to describe the setting, the protagonists and the main events of a news paper article. To this end, particular emphasis was given to named entities.

Keyphrases obtained are: **summarization, machine learning, linguistic processing, keyphrase extraction.**

Summary generated by direct extraction of sentences:

1.In section 2 we report on the general architecture of our system, which combines a **machine learning** approach with a **linguistic processing** of the document.
2.In this section we describe LAKE (Learning Algorithm for Keyphrase Extraction), a **keyphrase extraction** approach developed at ITC-irst based on a supervised learning approach that makes use of a **linguistic processing** of the documents.
3.2.A document is converted to a vector of features and **machine learning** techniques are used to induce a mapping from the feature space to the set of keyphrases (i.e. labels).
4.3.A feature vector is calculated for each candidate phrase and **machine learning** techniques are used to learn a model which classifies each candidate phrase as a keyphrase or a non-keyphrase.
5.4.The system works in two phases (see figure 1): first it considers a number of well motivated candidate keyphrases from a given document: then it uses a **machine learning** framework to select significant keyphrases for that document.
6.As our first participation in DUC, we developed LAKE, a system that exploits the role of **Keyphrase Extraction** as a useful approximation to **summarization**, evaluating its performance in task.

Summary generated by Mutual Reinforcement Principle:

1.As our first participation in DUC, we developed LAKE, a system that exploits the role of **Keyphrase Extraction** as a useful approximation to **summarization**, evaluating its performance in Task 1.
2.In section 2 we report on the general architecture of our system, which combines a **machine learning** approach with a **linguistic processing** of the document.
3.In this section we describe LAKE (Learning Algorithm for Keyphrase Extraction), a **keyphrase extraction** approach developed at ITC-irst based on a supervised learning approach that makes use of a **linguistic processing** of the documents.
4.There are two major tasks exploiting keyphrases: keyphrase assignment and **keyphrase extraction**. A document is converted to a vector of features and **machine learning** techniques are used to induce a mapping from the feature space to the set of keyphrases (i.e. labels).
5.In **keyphrase extraction** (KE), keyphrases are selected from the body of the input document, without a predefined list.
6.A feature vector is calculated for each candidate phrase and **machine learning** techniques are used to learn a model which classifies each candidate phrase as a keyphrase or a non-keyphrase.

Computing forward and backward mention for sentences in the resultant summary

| Sentence number | Backward count (B) | Forward Count (F) | No of keywords in a sentence (N) | Mean M= (B/N+F/N)/2 | Deviation D=abs(M-F) |
|---|---|---|---|---|---|
| 2. | 0 | 2 | 2 | 0.5 | 1.5 |
| 1. | 0 | 3 | 2 | 0.75 | 2.25 |
| 3. | 2 | 2 | 2 | 1 | 1 |
| 5. | 1 | 1 | 1 | 1 | 0 |
| 6. | 3 | 0 | 1 | 1.5 | 1.5 |
| 4. | 2 | 1 | 1 | 1.5 | 0.5 |

The resultant ordering of the sentences obtained is 2,1,3,5,6,4

**The output Sequence after computing the forward and backward mentions:**

2.In section 2 we report on the general architecture of our system, which combines a **machine learning** approach with a **linguistic processing** of the document.
1.As our first participation in DUC, we developed LAKE, a system that exploits the role of **Keyphrase Extraction** as a useful approximation to **summarization**, evaluating its performance in Task 1.
3.In this section we describe LAKE (Learning Algorithm for Keyphrase Extraction), a **keyphrase extraction** approach developed at ITC-irst based on a supervised learning approach that makes use of a **linguistic processing** of the documents.
5.A document is converted to a vector of features and **machine learning** techniques are used to induce a mapping from the feature space to the set of keyphrases (i.e. labels).

6.In **keyphrase extraction** (KE), keyphrases are selected from the body of the input document, without a predefined list.
4.There are two major tasks exploiting keyphrases: keyphrase assignment and **keyphrase extraction**.

## Example 3:

The contents of the input file:

Text **summarization** has become an important and timely tool for assisting and interpreting text information in today's fast-growing information age. There is an abundance of text material available on the Internet, however, usually the Internet provides more information than is needed. Therefore, a twofold problem is encountered: searching for relevant documents through an overwhelming number of documents available, and absorbing a large quantity of relevant information. **Summarization** is a useful tool for selecting relevant texts, and for extracting the key points of each text. Some articles such as academic papers have accompanying abstracts, which present their key points. However, news articles have no such accompanying summaries, and their titles are not sufficient to convey their important points. Therefore, a **summarization** tool for news articles would be extremely useful, since for a given news topic or event, there are a large number of available articles from the various new agencies and newspapers. Because news articles have a highly structured document form, important ideas can be obtained from the text simply by selecting sentences based on their attributes and locations in the article, We propose a machine learning approach that uses artificial neural networks to produce summaries of arbitrary length of news articles. A **neural network** is trained on a corpus of articles. The **neural network** is then modified, through **feature fusion**, to produce a summary of highly ranked sentences in the article. Through **feature fusion**, the network discovers the importance of various features used to determine the summary-worthiness of each sentence, the input to the **neural network** can be either real or binary vectors. There are two divergent approaches to automatic text summarization: 1) **summarization** based on abstraction, where the text has to be understood, and the summary produces from such an understanding, and 2) **summarization** based on extraction, which involves selecting a number of important sentences from the source text. **Summarization** by abstraction is concerned with issues related to text understanding, semantic representation and modification, and natural language processing. A review of the abstraction approach can be found in [1]. The first step in **summarization** by extraction is the identification of important features. There are two different types of features: non-structured features and structured features. One group of researches utilizes only non-structured features. On the other hand, a group of researches attempt to exploit structural relations between units of consideration. In our approach, we utilize a **feature fusion** technique to discover which features out of the available ones are actually useful, without manual intervention. There are three phases in our process: **neural network** training, **feature fusion**, and sentence selection. The first step involves training a **neural network** to recognize the type of sentences that should be included in the summary. The second step, **feature fusion**, prunes the **neural network** and collapses the hidden layer unit activations into discrete values with identified frequencies. This step generalizes the important features that must exist in the summary sentences by fusing the features and finding trends in the summary sentences. The third step, sentence selection, uses the modified neural network to filter the text and to select only the highly ranked sentences. This step controls the selection of the summary sentences in terms of their importance. These three steps are explained in detail in the next sections. The first phase of the process involves training the **neural network** to learn the types of sentences that should be included in the summary. The **neural network** learns the patterns inherent in sentences that should be included in the summary and those that should not be included. We use a three-layered feed forward neural network, which has been proven to be a universal function approximator. Once the network has learned the features that must exist in summary sentences, we need to discover the trends and relationships among the features that are inherent in the majority of sentences. This is accomplished by the **feature fusion** phase, which consists of two steps: 1) eliminating uncommon features 2) collapsing the effects of common features. Once the **pruning** step is complete, the network is trained with the same dataset in phase one to ensure that the recall accuracy of the network has not diminished significantly. If the recall accuracy of the network drops by more than 2%, the pruned connections and the neurons are restored and a stepwise **pruning** approach is pursued. In the stepwise **pruning** approach, the incoming and outgoing connections of the hidden layer neurons are pruned and the network is re-trained and tested for recall accuracy, one hidden layer neuron at a time. After **pruning** the network, the hidden layer activation values for each hidden layer neuron are clustered utilizing an adaptive clustering technique. Since dynamic clustering is order sensitive, the activation values are re-clustered. The radius of new clusters is set to one-half of the original clusters. The benefits of re-clustering are two-fold: due to order sensitivity of dynamic clustering, some of the activation values may be misclassified. Re-clustering alleviates this deficiency by classifying the activation values is appropriate clusters. Re-clustering with one-half of the original radius eliminates any possible overlaps among clusters. The combination of generalizing the effects of sentence features. Each cluster is identified by its centriod and frequency. **Feature fusion** phase provides control parameters, which can be used, for sentence making.

Keyphrases obtained are: **summarization, neural network, pruning, feature fusion**

Summary using direct extraction of sentences:

1.Once the **pruning** step is complete, the network is trained with the same dataset in phase one to ensure that the recall accuracy of the network has not diminished significantly.
2.If the recall accuracy of the network drops by more than 2%, the pruned connections and the neurons are restored and a stepwise **pruning** approach is pursued.
3.In the stepwise **pruning** approach, the incoming and outgoing connections of the hidden layer neurons are pruned and the network is re-trained and tested for recall accuracy, one hidden layer neuron at a time.
4.After **pruning** the network, the hidden layer activation values for each hidden layer neuron are clustered utilizing an adaptive clustering technique.
5.A **neural network** is trained on a corpus of articles.
6.The **neural network** is then modified, through **feature fusion**, to produce a summary of highly ranked sentences in the article.

Summary generated by Mutual Reinforcement Principle:

1.There are two divergent approaches to automatic text **summarization**: 1) **summarization** based on abstraction, where the text has to be understood, and the summary produces from such an understanding, and 2) **summarization** based on extraction, which involves selecting a number of important sentences from the source text.
2.The **neural network** is then modified, through **feature fusion**, to produce a summary of highly ranked sentences in the article.
3.There are three phases in our process: **neural network** training, **feature fusion**, and sentence selection.
4.The second step, **feature fusion**, prunes the **neural network** and collapses the hidden layer unit activations into discrete values with identified frequencies.
5.A **neural network** is trained on a corpus of articles.
6.The input to **neural network** can be either real or binary.

Computing forward and backward mentions for sentences in the resultant summary

| Sentence number | Backward count (B) | Forward Count (F) | No of keywords in a sentence (N) | Mean M= (B/N+F/N)/2 | Deviation D=abs(M-F) |
|---|---|---|---|---|---|
| 1. | 0 | 0 | 3 | 0 | 0 |
| 2. | 0 | 4 | 2 | 1 | 3 |
| 3. | 2 | 3 | 2 | 1.25 | 1.75 |
| 4. | 3 | 2 | 2 | 1.25 | 0.75 |
| 6. | 4 | 0 | 1 | 2 | 2 |
| 5. | 3 | 1 | 1 | 2 | 1 |

The resultant ordering of the sentences obtained is 1,2,3,4,6,5.

**The output Sequence after computing the forward and backward mentions:**

1.There are two divergent approaches to automatic text **summarization**: 1) **summarization** based on abstraction, where the text has to be understood, and the summary produces from such an understanding, and 2) **summarization** based on extraction, which involves selecting a number of important sentences from the source text.
2.The **neural network** is then modified, through **feature fusion**, to produce a summary of highly ranked sentences in the article.
3.There are three phases in our process: **neural network** training, **feature fusion**, and sentence selection.
4.The second step, **feature fusion**, prunes the **neural network** and collapses the hidden layer unit activations into discrete values with identified frequencies.
6.The input to **neural network** can be either real or binary.
5.A **neural network** is trained on a corpus of articles.

## 5.5 Discussion of Results

It is quite apparent from the above results that the summaries obtained by Mutual Reinforcement Principle are quite sensible, meaningful and consistent. The summaries generated using the mutual reinforcement principle are related to the main topic of the document as seen in the above examples. The summaries generated by direct extraction of sentences are good but not in all cases. The algorithm Kea, gives much importance to infrequently occurring terms i.e. highest score will be assigned to most rarely occurring terms. When we extract sentences based on the importance of these keyphrases there are chances that not much important and relevant sentences will be selected. Finally we end up in knowing nothing about the document's content. But this is not the case with reinforcement principle. Keyphrases occurring frequently i.e. in more number of sentences will be given much importance. Hence sentences selected will be containing frequently occurred keyphrases which give the overview of the document's contents.

For example, the input text in the example 3 has "pruning" as one of the keyphrases. But it has occurred only 4 times in the whole text and that too somewhere at the end of the text. According to the Kea algorithm, it has been assigned the highest score. Hence the summary obtained by direct extraction of sentences consists of all the sentences with the keyphase "pruning". But the document actually deals with "summarization". It's a bit misleading to the user i.e. the key content of the text could not be captured by the system. As discussed above the sentences extracted to form a summary are random and unrelated in nature. Inorder to get a cohesive summary internal links between the keyphrases have been considered. The performance of the method is quite satisfactory. The resultant summary has a good ordering of sentences and the summary is quite meaningful and cohesive.

Overall, it is encouraging that the mutual reinforcement principle followed by the computation of the internal links to obtain cohesive summaries gave substantially better performance compared to the straightforward method of direct extraction of sentences.

# Chapter 6
# Conclusion

The design of automatic text summarization technique is of great importance to the current world, which is filled with enormous amount of textual information. It would reduce the pain, people spend in reading the huge amounts of textual information by offering them a concise summary for each document and also saves their valuable time. The present work includes summarization of text documents using automatically extracted keyphrases. Kea algorithm, which is based on naïve bayes learning scheme, is used for automatic extraction of keyphrases. The summaries are generated by direct extraction of sentences and also using the technique of mutual reinforcement principle to generate a concise, sensitive, and a relevant summary, which captures the main topic. The performance of mutual reinforcement technique has been compared to that of direct extraction of sentences. The experimental results depict the summaries generated by mutual reinforcement principle are quite encouraging than that of direct selection of sentences. A sentence ordering scheme [20] to produce cohesive summaries using the internal links between keyphrases within the extracted sentences has also given satisfactory results. This ordering helps in generating brief summaries by selecting a few top sentences.

The input to the present system is plain text file. Further work is required to extend the system to handle various types of data like web pages, pdf files etc. The extracted summary may contain some unwanted phrases, which are of not much importance. Methods can be employed for automatically removing such extraneous phrases from sentences that are extracted as summary. The present work concentrates on summarization of single documents only. One of the important directions for further work would be to extend the system to multidocument summarization [2, 3].

# References

[1] Barzilay, R. and Elhabad, M. "Using lexical chains for text summarization" In Intelligent Scalable Summarization Workshop, ACL, 1997.

[2] Barzilay, R., Elhadad, N. and McKeown, K. 2001. "Sentence ordering in multidocument summarization." In proceedings of HLT 2001.

[3] Barzilay, R., "Information Fusion for Multidocument Summarization: Paraphrasing and Generation.", Ph.D. Thesis, Columbia University, 2003.

[4] Chuang, W. T. and Yang, J. "Extracting sentence segments for text summarization: A machine learning approach" Proceedings of the 23rd International Conference on Research in Information Retrieval (SIGIR '00), pp. 152-159, 2000.

[5] Ernesto D'Avanzo, Bernardo Magnini, Alessandro Vallin. "Keyphrase Extraction for summarization purposes: The LAKE system at DUC-2004" Document Understanding Conference (DUC), edited by Paul Over, Boston, MA, USA, 6-7 May 2004.

[6] Fayyad, U.M., Platestsky-Shapiro, G. and Smyth, P. *"Advances in Knowledge Discovery and Data Mining"*, chapter From Data Mining to Knowledge Discovery: An Overview, pages 1--30. AAAI Press, Menlo Park, CA, 1996.

[7] Frank, E., Paynter, G.W., Witten, I.H., Gutwin, C. and Nevill-Manning, C.G. (1999) "Domain-specific Keyphrase Extraction" *Proc. Sixteenth International Joint Conference on Artificial Intelligence*, Morgan Kaufmann Publishers, San Francisco, CA, pp. 668-673.

[8] Guo, Y., Stylios, G. K., "An Intelligent Algorithm for Automatic Document Summarisation" IEEE International Conference on Natural Language Processing and Knowledge Engineering, (NLP-KE 2003), Beijing, China, (October 2003).

[9] Han, J., Kamber,. M., "Data Mining: Concepts and Techniques", The Morgan Kaufmann Series in Data Management Systems, Jim Gray, Series Editor Morgan Kaufmann Publishers, August 2000. 550 pages. ISBN 1-55860-489-8

[10] Hahn, Udo (1985). "On lexically distributed text parsing. A computational model for the analysis of textuality on the level of text cohesion and text coherence." In Linking in text, edited by Ferenc Kiefer, University of Konstanz.

[11] Halliday 1985 : Halliday, M.A.K., "An introduction to functional grammar." London ; Baltimore, Md., USA : Edward Arnold, 1985.

[12] Hearst, M., "Untangling Text Data Mining,"in the Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics, 1999.

[13] Hongyan Jing and Kathleen R. McKeown. 2000. "Cut and paste based text summarization." In Proceedings of NAACL 2000.

[14] Hongyuan Zha. "Generic summarization and keyphrase extraction using mutual reinforcement principle and sentence clustering" Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, (SIGIR '02), pp. 113-120, 2002.

[15] Ingrid Renz , Jurgen Franke. "Text Mining." Physica-Verlag Publication 2003.

[16] Kupiec, J., Pederson, J. and Chen, F., "A Trainable Document Summarizer" Proceedings of the 18th International Conference on Research in Information Retrieval (SIGIR '95), pp. 55-60, 1995.

[17] Michael White, Tanya Korelsky, Claire Cardie, Vincent Ng, David Pierce and Kiri Wagstaff. 2001. "Multidocument Summarization via Information Extraction" Proceedings of HLT 2001, Human Language Technology Conference. San Diego, CA

[18] Sparck-Jones, K. "Automatic summarizing: factors and directions." In Mani, I.; Maybury, M. *Advances in Automatic Text Summarization.* The MIT Press (1999) 1-12

[19] Steve Jones, Stephen Lundy, Gordon W. Paynter. " Interactive Document Summarization Using Automatically Extracted Keyphrases." Proceedings of 35th Hawaii International Conference on System Sciences-2002.

[20] Parul Luthra, "Automatic Text Summarization", M.Tech(Computer Applications) Dissertation 2005, Department of Mathematics, IIT Delhi.

[21] Pujari, A. K., "Data mining techniques ", University Press Ltd, Hyderabad, India ISBN-8173713804, February 2001.

[22] Turney, P.D., "Learning to extract keyphrases from text." Technical Report ERB-1057. NRC #41622,National Research Council, Institute for Information Technology, 1999.

[23] Turney, P.D., "Learning algorithms for keyphrase extraction." Submitted to *Information Retrieval*-INRT 34-99.

[24] Turney, P. D., "Mining the web for lexical knowledge to improve keyphrase extraction: Learning from labeled and unlabeled data." NRC/ERB-1096. July 19, 2002. 32 pages NRC 44947.

[25] Witten, I.H., Paynter, G.W., Frank, E., Gutwin, C. and Nevill-Manning, C.G. (1999) "KEA:Practical automatic keyphrase extraction"*Proc. DL '99*, pp. 254-256. (Poster presentation.)