ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 10, October 2015

Email Summarization-Extracting Main Content from the Mail

Mubashir Alam[1] ,Mohit Kakkar[2]

[1] M.Tech Student, Dept. of CSE, Desh Bhagat University, Mandi Gobindgarh, Punjab, India

[2]Assistant Professor, Dept. of CSE, Desh Bhagat University, Mandi Gobindgarh, Punjab, India

ABSTRACT:A summary of document is a shorter text conveys the most important information from the sources .Summary of the text must contains important information from the documents. This paper presents the design and implementation of a system to summarize e mail messages. The system uses the subject and contents of the e mail message to classify e mails based on user's activities and generate summary of each incoming message.

KEYWORDS: Email, Email summarization, Summarization , Incoming messages, Spam filtering

I.INTRODUCTION

A summary can be defined as a text that is produced from one or more texts, that contain a significant portion of the information in the original text(s), and that is no longer than half of the original text(s). According to, text summarization is the process of distilling the most important information from a source (or sources) to produce an abridged version for a particular user (or user) and task (or tasks). Summarization is the process of reducing a text document in order to create a summary that retains the most important points of the original document. As the problem of information overload has grown, and as the quantity of data has increased, so has interest in automatic summarization. Technologies that can make a coherent summary take into account variables such as length, writing style and syntax. An example of the use of summarization technology is search engines such as Google.With the ever increasing popularity of emails, email overload becomes a major problem for many email users. Users spend a lot of time reading, replying and organizing their emails. To help users organize their email folders, many forms of support have been proposed, including Spam filtering, email classification and email visualization. In this paper, we discuss a different form of Support email summarization. The goal is to provide a concise, informative summary of emails contained in a folder, thus saving the user from browsing through each email one by on. Email summarization can also be valuable for users reading emails with mobile devices. Given the small screen size of handheld devices, efforts have been made to redesign the user interface. However, providing a concise summary may be just as important.

II.RELATED WORK

One of the common existing methods is to manually archive messages into folders with a view of reducing the number of information objects a user must process at any given time. However, this is an insufficient solution as folder names are not necessarily a true reflection of their content and their creation and maintenance can impose a significant burden on the user [1].There are several examples of email classifiers that attempt to sort out mails into folders, semi-automatically such as:Ishmail [3]: It automatically sorts email messages into folders and orders them by importance.

• Commercial email clients [4]: Most popular commercial email clients like Procmail, Eudora, Mozilla Thunderbird, Microsoft Outlook and Outlook Express also support message filing according to user defined rule sets.

• IBM's MailCat [5]: It adapts dynamically to observed users' mail-filling habits and provides a list of three folders most likely to be appropriate for a given message.

• Magi [6]: This system records each email interaction and uses a learning algorithm to classify new messages based on the user's prior behavior.

Jan Ulrich, Giuseppe Carenini , Gabriel Murray and Raymond Ng,2009,Regression-Based Summarization [03].

In this paper we present a regression-based machine learning approach to email thread summarization. The regression model is able to take advantage of multiple gold-standard annotations for training purposes, in contrast to most work with binary classifiers. We also investigate the usefulness of novel features such as speech acts. This paper also introduces a newly created and publicly available email corpus for summarization research. We show that regression-based classifiers perform better than binary classifiers because they preserve more information about annotator judgments. Inour comparison between different regression-based classifiers, we found that Bagging and Gaussian Processes have the highest weighted recall.

Giuseppe Carenini, Raymond T. Ng, Xiaodong Zhou, 2007,Summarizing Email Conversations with Clue Words[07].

In this paper, we propose a new framework for email summarization. One novelty is to use a fragment quotation graph to try to capture an email conversation. The second novelty is to use clue words to measure the importance of sentences in conversation summarization. Based on clue words and their scores, we propose a method called CWS, which is capable of producing a summary of any length as requested by the user. We provide a comprehensive comparison of CWS with various existing methods on the Enron data set. Preliminary results suggest that CWS provides better summaries than existing methods.

Vishal Gupta, 2010, A Survey of Text Summarization Extractive Techniques[04].

This survey paper is concentrating on extractive summarization methods. An extractive summary is selection of important sentences from the original text. The importance of sentences is decided based on statistical and linguistic features of sentences. Many variations of the extractive approach have been tried in the last ten years. However, it is hard to say how much greater interpretive sophistication, at sentence or text level, contributes to performance. Without the use of NLP, the generated summary may suffer from lack of cohesion and semantics. If texts containing multiple topics, the generated summary might not be balanced. Deciding a proper weight of individual features is very important as quality of final summary is depending on it. We should devote more time in deciding feature weights.

Kirti Bhatia, Dr. Rajendar Chhillar, 2014,A Statistical Approach to perform Web Based Summarization[05].

This research focuses on developing a statistical automatic text summarization approach, K-mixture probabilistic model, to enhancing the quality of summaries. Sentences are ranked and extracted based on their semantic relationships significance values. The objective of this research is thus to propose a statistical approach to text summarization.

In this present work we have defined feature based evaluation approach to perform the document summarization. We have connected the work with web page extraction. In the feature phase, the statistical information is being extracted to perform the summarization.

III.PROBLEM FORMULATION

One of the common existing methods is to manually archive messages into folders with a view of reducing the number of information objects a user must process at any given time. However, this is an insufficient solution as folder names are not necessarily a true reflection of their content and their creation and maintenance can impose a significant burden on the user.

There are several examples of email classifiers that attempt to sort out mails into folders, semi-automatically such as:

• Ishmail: It automatically sorts email messages into folders and orders them by importance.

• Commercial email clients: Most popular commercial email clients like Procmail,
Eudora, Mozilla Thunderbird, Microsoft Outlook and Outlook Express also support message filing according to user defined rule sets.

• IBM's MailCat [5]: It adapts dynamically to observed users' mail-filling habits and provides a list of three folders most likely to be appropriate for a given message.
• Magi [6]: This system records each email interaction and uses a learning algorithm to classify new messages based on the user's prior behavior
Email summarization can be viewed as a special case of multi-document (MD) summarization. Radev et al. develop MEAD which gives a score to each sentence based on its similarity to the TFIDF centroid of the whole document set and other properties such as position in a document, sentence length and inter-sentence similarity. Erkan etal. develop the LexPageRank to rank sentences based on the eigenvector centrality. They compute a sentence link-age matrix as the sentence similarity and use this matrix with the well-known PageRank algorithm. Wan et al.generate an affinity graph from multiple documents and use this graph for summarization. They consider both the in-formation richness and the sentence novelty based on the sentence affinity graph. However, MD summarization methods, when applied to emails, do not take into account the key differences between emails and conventional documents. Key differences include the referential structure of conversations, the existence of hidden emails and the high variability of writing styles. Section 6 will compare CWS with MEAD for email summarization. Rambow et al. apply a machine learning approach to email summarization. They use RIPPER as a classifier to determine which sentences should be included in a summary. Features used for learning include linguistic features, and features describing the email and the threading structure. Such an approach requires a large number of positive examples and cannot produce summaries with varying length based on the users request. It is also not clear how this approach can handle hidden emails. Section 6 will com-pare CWS with RIPPER.Wan et al. study decision-making summarization for email conversations. Email threading is used. Among the various sets of features explored, their experiments show that a centroid based method is effective. In our earlier studies, we focus on the re-construction of hidden emails. The focus here is completely different in generating summaries of conversations, regardless of whether there are hidden emails or not. In, we use a precedence graph to re-construct hidden emails. The fragment quotation graph here is different in at least two ways. First, the nodes are different as a fragment quotation graph creates nodes for both new and hidden fragments. More importantly, the edges in a precedence graph capture textual ordering of the nodes within one hidden email, whereas the edges in a fragment quotation graph reflect the referential relationship among multiple emails. As for extracting conversations, Yeh et al. study how to use quotation matching to construct email threads. Their experiments show a higher recall than the header-based threading method. This supports our use of the quotation graph as a representation of email conversation. Shrestha et al. propose methods to automatically identify the question-answer pairs from an email thread. Their method may be useful in building the conversation structure for the purpose of email summarization. Agrawal et al. extract social networks from newsgroups. Stolfo et al. study the behaviour model of email users based on the social network analysis among email correspondences. They develop an email mining toolkit and use it to identify target emails without analysing the email content.We have designed and developed a system that summarizes email messages and also groups emails into activities.

IV.OBJECTIVE

Summarization is a hard problem of Natural Language Processing because, to do it properly, one has to really understand the point of a text. This requires semantic analysis, discourse processing, and inferential interpretation (grouping of the content using world knowledge). The last step, especially, is complex, because systems without a great deal of world knowledge simply cannot do it. Therefore, attempts so far of performing true abstraction--creating abstracts as summaries--have not been very successful. Fortunately, however, an approximation called extraction is more feasible today. To create an extract, a system needs simply to identify the most important/topical/central topic(s) of the text, and return them to the reader. Although the summary is not necessarily coherent, the reader can form an opinion of the content of the original. Most automated summarization systems today produce extracts only. Summarist is an attempt to develop robust extraction technology as far as it can go and then continue research and development of techniques to perform abstraction. This work faces the depth vs. robustness tradeoff: either system analyze/interpret the input deeply enough to produce good summaries (but are limited to small application domains), or they work robustly over more or less unrestricted text (but cannot analyze deeply enough to fuse the input into a true summary, and hence perform only topic extraction). In particular, symbolic techniques, using parsers, grammars, and semantic representations, do not scale up to real-world size, while Information Retrieval and other statistical techniques, being based on word counting and word clustering, cannot create true summaries because they operate at the word (surface)

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 10, October 2015

level instead of at the concept level.To date, summarist produces extract summaries in five languages (and has been linked to translation engines for these languages in the Must system). Work is underway both to extend the extract-based capabilities of summarist and to build up the large knowledge collection required for inference-based abstraction.

V.METHODODLOGY

The steps included in the methodology are given as

- The System will first parses the query language in natural language and finds the major parts in the string.
- Then first it will look for the table and then it parses the string.
- After parsing it will construct the parse tree of the abstracted symbols.
- Once the parse tree is generated will analyze the prioritization and the frequency of the abstracted symbols.
- All these symbols and keywords will be documented in a table.
- Now we will analyze the user requirement of summarization.
- Finally we will extract all the sentences having the same keywords respective to the priority and the user requirement.
- Analysis-Final step of research will be analyzed.

Email Summarizer

In this project we have developed an email summarizer. This summarizer summarizes the emails and also extract dates from an email in order to make sure some important info like event, meeting, anniversary or birthday etc. is captured as important attribute. Let us have a look at the overall working on the system.

The system consists of four Java files, each file with its own working. These files are as follow:

1. Main file (EmailSummarizer.java)
2. Tokenizer.java
3. Summarizer.java
4. PorterStemmer.java

Email Summarizer

This file is the main entry point of the program. The program starts and presents a simple prompt to select a choice. These choices are: Enter a paragraph manually or Enter Paragraph from a file.

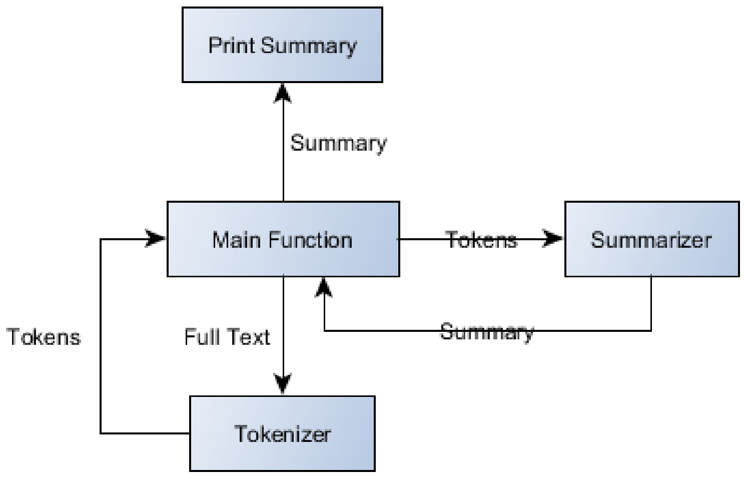


Fig 1Email summarization

After the option selection, the system asks if we want to save the summary to a file (output to a file). Then the system uses the Tokenizer.java class to tokenize all the keywords from the text paragraph and then it processes it and generates a summary using Summarizer.java.

Tokenizer

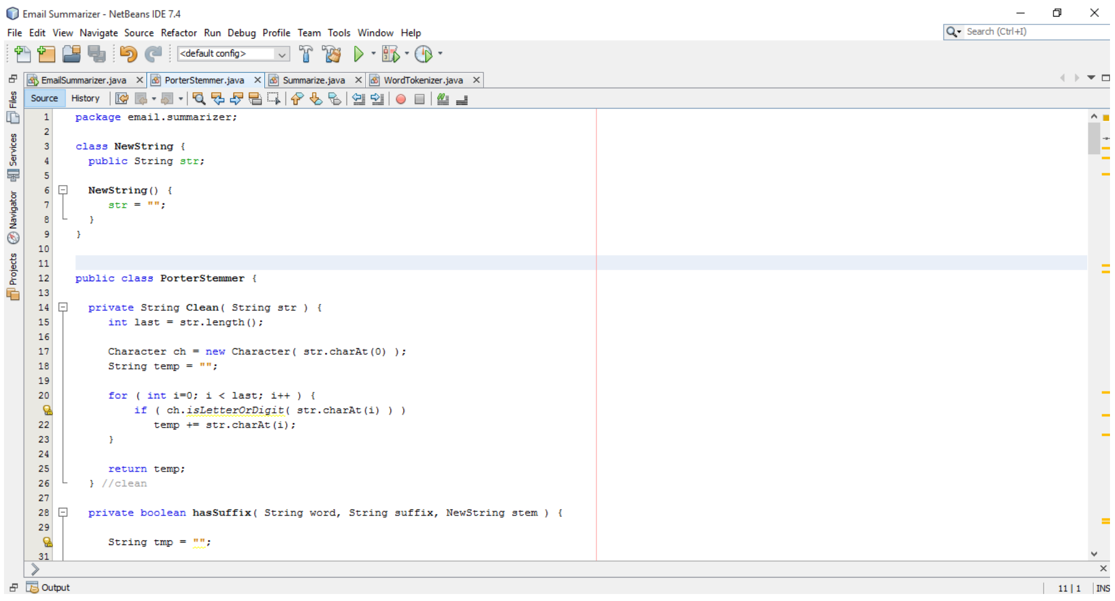
This file presents a complete solution for tokenizing the keywords. The system starts by breaking sentences by finding appropriate full stops in the file. This system is intelligent enough to not follow decimal points and abbreviations.
Once the sentences are extracted, the system finds all the words in the sentence. After finding all the words, the system runs a procedure to filter all the stop words of English language in order to present more important keywords only. The system then returns these keywords as tokens.

Summarizer

This file is the heart of the system. This file starts off by getting all the keywords or tokens generated by Tokenizer.java class. It then extracts dates from these keywords. This is an important step. Once all the dates are extracted it stems the tokens using PorterStemmer.java class. After that, the system runs a check to find frequency of the tokens and extract top 4 tokens. The date token is preserved. After that, the system finds all the sentences that contain those tokens and then filters out top 5 sentences and writes these sentences in an order to generate summary. If some date tokens were found, the tokens are filtered to present a sentence on top mentioning the data.

VI.RESULTS AND EXPERIMENT

The system is developed in JAVA programming language and is tested on different text pieces. Since, the analysis of any summarizing system is qualitative only, so the qualitative analysis is performed and results are shown.
The figure 2 shows the main entry point code, this code actually instantiate other classes like Tokenizer and Summarizer which actually performs summarization of the text.



```java
package email.summarizer;

class NewString {
  public String str;

  NewString() {
     str = "";
  }
}


public class PorterStemmer {

  private String Clean( String str ) {
     int last = str.length();

     Character ch = new Character( str.charAt(0) );
     String temp = "";

     for ( int i=0; i < last; i++ ) {
         if ( ch.isLetterOrDigit( str.charAt(i) ) )
            temp += str.charAt(i);
     }

     return temp;
  } //clean

  private boolean hasSuffix( String word, String suffix, NewString stem ) {

     String tmp = "";
```

Figure 2 Screenshot of Java code in Netbeans IDE (Porter Stemmer Algorithm)

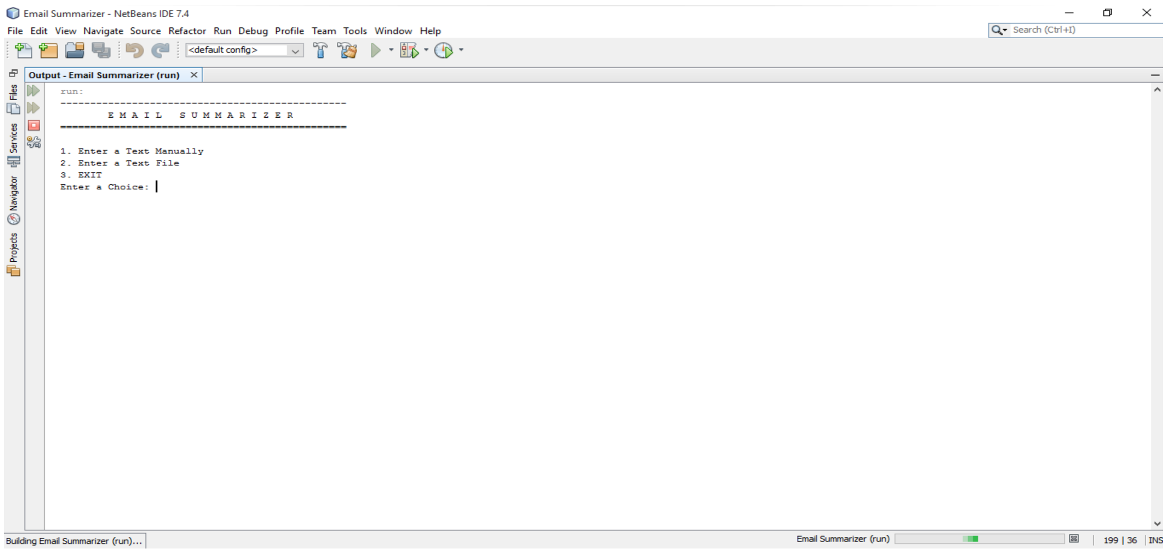
ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 10, October 2015

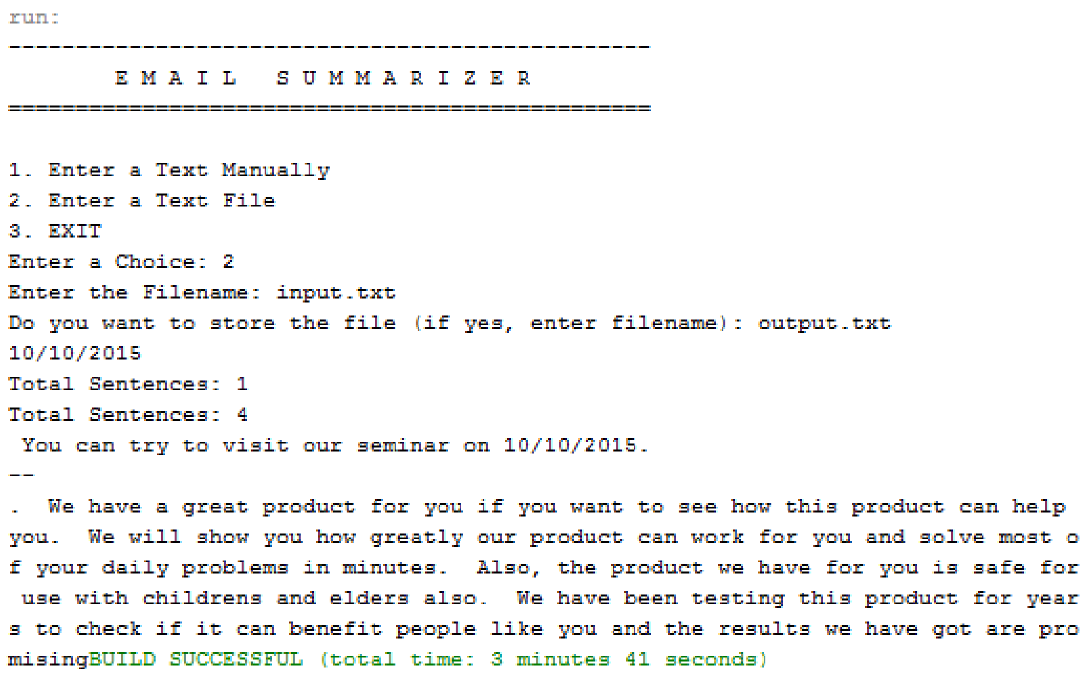
The figure 2 shows the porter stemmer class. This class actually performs word stemming after the word tokenized. This class is instantiated in Tokenizer class.


Email Summarizer - NetBeans IDE 7.4
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Search (Ctrl+I)
<default config>
Output - Email Summarizer (run)
Files
Services
Navigator
Projects
Building Email Summarizer (run)...
Email Summarizer (run)
199 | 36
INS

```
run:
------------------------------------------------
       E M A I L   S U M M A R I Z E R
================================================

1. Enter a Text Manually
2. Enter a Text File
3. EXIT
Enter a Choice: 
```

This above fig shows the result by running the code first screen. It allows user to select two options or exit. It can allow user to type the text for summarization or input the file to summarize



```
run:
------------------------------------------------
       E M A I L   S U M M A R I Z E R
================================================

1. Enter a Text Manually
2. Enter a Text File
3. EXIT
Enter a Choice: 2
Enter the Filename: input.txt
Do you want to store the file (if yes, enter filename): output.txt
10/10/2015
Total Sentences: 1
Total Sentences: 4
 You can try to visit our seminar on 10/10/2015.
--
. We have a great product for you if you want to see how this product can help
you. We will show you how greatly our product can work for you and solve most o
f your daily problems in minutes. Also, the product we have for you is safe for
 use with childrens and elders also. We have been testing this product for year
s to check if it can benefit people like you and the results we have got are pro
misingBUILD SUCCESSFUL (total time: 3 minutes 41 seconds)
```

Figure 6: The result of summarization on a sample email.

The result of summarization on a sample email. The result also shows the extracted Date of Event and highlighted it by putting it on top.

VII.CONCLUSION & FUTURE SCOPE

In this thesis we have studied the concept of text summarization using Natural Language Processing. The text summarization is a complex subject and many challenges are there in order to effectively summarize the text. Many modern text summarizers only extract top keywords in form of summarization.

In our work, we used the top keywords and their combinations to extract sentences which have the same keywords or the combination of keywords (not strictly consecutively). These extracted sentences are concatenated in same order as they appear in the original text to achieve a better and effective text summarizer. **Future Scope:**

Future Scopeof this project is that we can also extract place from the sentences in where we have found date and thus making it more efficient. Also, we can add Named Entity Extraction to extract some common names like a product name, place name etc. We can also classify e-mail into groups.

REFERENCES

1. en.wikipedia.org/wiki /Natural Language Processing

2. https://www.cs.utexas.edu/~mooney/cs343/slide-handouts/nlp.pdf .

3. Jan Ulrich , Giuseppe Carenini , Gabriel Murray and Raymond Ng,2009,Regression-Based Summarization of Email Conversations,"Proceedings of the Third International ICWSM Conference (2009)"

4. Vishal Gupta,2010 ,A Survey of Text Summarization Extractive Techniques,"JOURNAL OF EMERGING TECHNOLOGIES IN WEB INTELLIGENCE, VOL. 2, NO. 3, AUGUST 2010".

5. Kirti Bhatia, Dr. Rajendar Chhillar, 2014,A Statistical Approach to perform Web Based Summarization,"IOSR Journal of Computer Engineering (IOSRJCE) ISSN : 2278-0661 Volume 1, Issue 6 (Aug-July 2012), PP 01-03 www.iosrjournals.org".

6. Elena Lloret, 2012,"TEXT SUMMARISATION:AN OVERVIEW".

7. Giuseppe Carenini, Raymond T. Ng, Xiaodong Zhou, 2007,Summarizing Email Conversations with Clue Words".http://www2007.org/papers/paper631.pdf

8. Taiwo Ayodele,Rinat Khusainov ,David Ndzi,2012, Email Classification and Summarization: A Machine Learning Approach

9. Vishal Patil, Mahalakshmy Krishnamoorthy, Parag Oke, Prof. M. Kiruthika,2012,A Statistical Approach for Document Summarization,"International Journal of Advanced Computer Technology (IJACT) ISSN:2319-7900".

BIOGRAPHY

The author name is mubashir alam.He has done B.Tech in Information Technology Engineering from Baba Ghulam Shah BadshahUniversity,Rajouri,J&K with firdt grade. Currently he is pursuing M.Tech in Computer Science Engineering from Desh Bhagat University. Mandi Gobindgarh, Punjab.The author has knowledge of working on various platform such as ASP.NET,JAVA,PHP.His research intrest are network security,biometrics,cryptoghaphy.
.