6-1-2014

Utilizing graph-based representation of text in a hybrid approach to multiple documents summarization

May Sayed Mahmoud

THE AMERICAN UNIVERSITY IN CAIRO

SCHOOL OF SCIENCES AND ENGINEERING

Utilizing Graph-based Representation of Text in a Hybrid Approach to Multiple Documents Summarization

A Thesis submitted to
Department of Computer Science and Engineering
In partial fulfillment of the requirements for the degree of
Master of Computer Science

By: May Abdelreheem Sayed
B.S., Computer Science
The American University in Cairo
May 2014

Approval

Has been approved by

Dr. Ahmed Rafea
Thesis Advisor ______________________________
Affiliation ______________________________

Dr. Mikhail Mikhail
Thesis Committee Member ______________________________
Affiliation ______________________________

Dr. Moahmed Mostafa
Thesis Committee Member ______________________________
Affiliation ______________________________

Dr. Ayman El-Dessoki
External Examiner ______________________________
Affiliation ______________________________

____________ /	____________		____________ /	____________
Department Chair	Date		Dean	Date

Dedication

I would like to dedicate this thesis to my mother for all the unconditional love and support she has given me throughout my life without which I would have aspired for nothing. I am also thankful for my father’s love, support and all the hard work he has put to give me the best chances in life. To my husband and son, you are the reason I work hard every day to be a success. Lastly, I want to dedicate my work to the loving memory of my Grandfather Abdelrahman who always believed in me.

Acknowledgment

I am very thankful to all the great people I have in my life who helped me finish my thesis. I would like to express my sincere gratitude to my supervisor Dr. Ahmed Rafea for all his guidance, support and advice throughout my thesis. I would like to also thank my thesis committee for their feedback.

I am also very grateful for all the guidance and knowledge I gained from my professors at AUC and for the support of the whole computer science department staff.

I would like to thank Balsam Amin, Sarah Nadi, Karim Hamdan and Mostafa Hamza for all their support and volunteering efforts to help in the experiments. I would also like to give special thanks to Balsam Amin for helping me proof-read my thesis and for all her excellent advice and support. I would also like to thank my friend and study partner Daniah Mokhtar, for all the time we spent together supporting each other during our journey in graduate studies.

Lastly, I want to thank my dear husband for all the endless support he has given me throughout my studies. Thank you for not letting me give up when I wanted to. Thank you for pushing me to excel and believing in me even when I doubted myself.

TABLE OF CONTENTS

List of Figures

List of Tables

ABSTRACT

The aim of automatic text summarization is to process text with the purpose of identifying and presenting the most important information appearing in the text. In this research, we aim to investigate automatic multiple document summarization using a hybrid approach of extractive and "shallow" abstractive methods. We aim to utilize the graph-based representation approach proposed in [1] and [2] as part of our method to multiple document summarization aiming to provide concise, informative and coherent summaries. We start by scoring sentences based on significance to extract top scoring ones from each document of the set of documents being summarized. In this step, we look into different criteria of scoring sentences, which include: the presence of highly frequent words of the document, the presence of highly frequent words of the set of documents and the presence of words found in the first and last sentence of the document and the different combination of such features. Upon running our experiments we found that the best combination of features to use is utilizing the presence of highly frequent words of the document and presence of words found in the first and last sentences of the document. The average f-score of those features had an average of 7.9% increase to other features' f-scores. Secondly, we address the issue of redundancy of information through clustering sentences of same or similar information into one cluster that will be compressed into one sentence, thus avoiding redundancy of information as much as possible. We investigated clustering the extracted sentences based on two criteria for similarity, the first of which uses word frequency vector for similarity measure and the second of which uses word semantic similarity. Through our experiment, we found that the use of the word vector features yields much better clusters in terms of sentence similarity. The word feature vector had a 20% more number of clusters labeled to contain similar sentences as opposed to those of the word semantic feature. We then adopted a graph-based representation of text

proposed in [1] and [2] to represent each sentence in a cluster, and using the k-shortest paths we found the shortest path to represent the final compressed sentence and use it as a final sentence in the summary. Human evaluator scored sentences based on grammatical correctness and almost 74% of 51 sentences evaluated got a perfect score of 2 which is a perfect or near perfect sentence. We finally propose a method for scoring the compressed sentences according to the order in which they should appear in the final summary.

We used the Document Understanding Conference dataset for year 2014 as the evaluating dataset for our final system. We used the ROUGE system for evaluation which stands for Recall-Oriented Understudy for Gisting Evaluation. This system compare the automatic summaries to “ideal” human references. We also compared our summaries ROUGE scores to those of summaries generated using the MEAD summarization tool. Our system provided better precision and f-score as well as comparable recall scores. On average our system has a percentage increase of 2% for precision and 1.6% increase in f-score than those of MEAD while MEAD has an increase of 0.8% in recall. In addition, our system provided more compressed version of the summary as opposed to that generated by MEAD. We finally ran an experiment to evaluate the order of sentences in the final summary and its comprehensibility where we show that our ordering method produced a comprehensible summary. On average, summaries that scored a perfect score in term of comprehensibility constitute 72% of the evaluated summaries. Evaluators were also asked to count the number of ungrammatical and incomprehensible sentences in the evaluated summaries and on average they were only 10.9% of the summaries sentences. We believe our system provide a “shallow” abstractive summary to multiple documents that doesn’t require intensive Natural Language Processing.

CHAPTER 1. INTRODUCTION

1.1 OVERVIEW

With the current widespread use of the internet, we are bombarded by an influx of information from a multitude of sources that results in an information overload problem. You try to search for a topic online and you retrieve thousands of results from search engines to look through. And with today’s busy lifestyle, people are very pressed in time, with a strong tendency to seek quick and short answers to their information quests, effectively and efficiently. Accordingly, the need for automatic summarization is becoming of great value each day. In fact, we already refer to various forms of summarized content in our daily lives; from article reviews, abridged stories, news briefs, digests, meeting minutes and many other forms of summaries. Everyday, we tend to make a lot of decisions that require some form of prior information seeking; for example, “should I read this book?”, “should I watch this movie?”, “should I read this article?”, “what are the news highlights?”, “is it safe to travel to a certain area?”..etc. In addition, we tend to use our smart phones and mobile devices to do things like sending e-mails and browsing the internet. Due to their limited interfaces, summarization will become of great value to mobile devices providing the most important and relevant information to the user in a concise fashion [3].

Text Summarization is the process by which the most important information from a source is identified and distilled to produce a concise version of the original text [4]. Summaries could be of generic natures that are aimed for a wider range of audience or the user could tailor it to a specific query. Also summaries differ according to their intended use. Indicative summaries are used to indicate the topics discussed in the source text. Informative summaries are intended to present the concepts covered by the source text while evaluative or critical summaries offer a critique of the source [4]. Also, the form of the summary could be extractive, where it consists entirely of text extracted from the source text, or abstractive, where new material is introduced and the resulting summary becomes an abstract of the source text [4] , [5].

The history of automatic text summarization started in the late 1950's where focus on surface-level approach was examined in summarization. Entity level approach, which builds an internal representation for text, modeling text entities and their relationships were introduced in the early 1960's followed by other approaches introduced in the 1970's, like use of cue phrases. Currently, there is an interest in the field of summarization focusing mainly on extractive methods rather than abstractive ones as well as the emergence of a new focus of summarization like multimedia summarization and multilingual summarization [4].

The process of text summarization differs from one approach to the other. However most approaches require some sort of initial analysis to the text, followed by the transformation of the text to a summary representation, finally leading up to the synthesis of the appropriate output. They also include a basic condensation operation of selecting the most important information from text, aggregating the information from different parts and/or sources and finally generalizing information by substituting specific information with more abstract ones [4].

Nowadays, the field of automatic summarization is becoming more popular with the availability of more data and innovative methods in natural language processing. Also, new areas for applying summarization are becoming increasingly relevant; for example, the area of multilingual summarization as well as multimedia summarization that leverage cross-media information during the various phases of summarization [6], [4].

1.2 Problem Definition

The challenge of multiple documents summarization lies in selecting salient information, handling redundancy in information presented in the documents, and synthesizing the final summary. Most of the literature for summarization focuses on extractive methods, with very limited research that uses true abstractive methods. Extractive methods extract sentences as they appear in the articles and order them into a final summary. In [1] and [2], a graph-based representation of text is utilized to provide sentence compression and summarization of highly redundant opinions respectively. This is a form of "shallow" abstractive method used to provide final sentences that are not entirely the same form as the exact sentences appearing in the original text. This approach, to the best of our knowledge, has never been used in a full multiple document summarization to provide a full summary of several documents. We will research the use of the graph-based representation of text as part of our hybrid approach for solving the challenges of automatic multiple documents summarization.

1.3 Motivation

Today we are all faced with the problem of information overload, and with our busy lifestyles, we are very pressed in time to find the information we seek. Summarization provides an effective and efficient way for us to find the information we require to accomplish the tasks we are given in a short amount of time. With the widespread of smart phones and mobile devices, the methods of computing are changing, where people conveniently tend now to accomplish a lot of the tasks they used to do on their computer through their mobiles, like sending e-mails, logging on to their social networks, reading news and browsing the internet. However, mobile devices have limited screen space and limited interface capabilities, which makes it more important to compactly present the important information to the user. This is where summarization becomes of great value. We want to acquire information without having to go through multiple sources discussing the same topic. This is the primary motivation behind our interest in tackling the problem of summarization and trying to develop a refined method for summarization of multiple documents.

1.4 Objective

The main objective of this research is to tackle the problem of multiple documents summarization. We aim to utilize the graph-based approach proposed in [1], [2] in a hybrid method to provide a summary to multiple documents of the same topic. This hybrid method will contain extractive as well as "shallow" abstractive methods. We intend to address the challenges of selecting significant sentences and handling redundancy of information found in several documents, in addition to designing a method to order the final summary into a coherent entity.

1.5 Approach

Our approach started with literature review of the work done in automatic text summarization. This helps us understand the field and what problems were tackled before, along with the proposed approaches in the literature. This guided us to plan our approach for answering our research questions as well as understand how to evaluate our proposed system. We aim to build a framework to help us answer the following research questions for our proposed hybrid approach to summarize multiple documents. These main questions are:

- What work was done in the literature?
- What is the best method to use for selecting significant sentences in the extractive part of the approach?

- What is the method to use when clustering sentences based on similarity?
- How to use graph-based representation as a form of “shallow” abstractive approach?
- How to order the sentences into a final summary?
- How to evaluate our system?

Starting with the question of selecting significant sentences, we research the published methods in the literature and select the most commonly used methods that provide best results. We conduct an experiment for different features adopted from the literature; the first one selects sentences based on term frequency, the second uses clusters key words, and the third uses sentence position; and then we compare the automatically generated summaries from each feature (combination of features) to human reference summaries through ROUGE scores and find the best method to use in our approach. In order to handle redundancy found in selected sentences, we cluster them based on similarity. In this step, we researched two methods for clustering sentences; first by word term frequency matrix, as well as by word semantics adopted from [7]. We manually compare the produced clusters and select the method that provides the best cluster of sentences. After that, we use the graph-based representation of each cluster of sentences to provide a compressed one or, at most, two sentences of the cluster. Finally, we propose a new method to score the sentences produced from the previous steps to provide an ordering score that will be used to order the sentences to produce the final summary. We used the dataset from Document Understanding Conference 2004 that consists of newswire articles clustered by events or topic. We compared our work to MEAD system [8] using the ROUGE evaluation system. We also evaluated the order of sentences in our produced summaries using human evaluator as compared to those summaries produced by MEAD.

1.6 THESIS LAYOUT

Chapter two of the thesis document consists of a thorough literature review of the work done on automatic text summarization. The review starts with document summarization, presenting work done in single document summarization, followed by work done in multiple documents summarization. It then addresses work done in web page summarization, finally showing some other form of automatic text summarization presented in the literature. Chapter three details our research methodology and the different research experiments we conducted. In chapter four, we present our system design and the different aspects of the summarization process. In chapter five, we summarize our final system evaluation starting with a description of the dataset used and description of the MEAD summarization tool which we used as a baseline to compare our system to. We then present our results for the final multiple document summarization of the dataset. We present the ROUGE-1, ROUGE-2 and ROUGE-SU4 of our system and that of the MEAD system and present a comparison between them. We end the chapter with the experiment we conducted to evaluate the ordering criteria used to order sentences in the generated summaries. Finally, in chapter six, we present our conclusion and future work. In the end we present the references we used throughout the document followed by appendix A that contains some sample summaries generated by our system.

CHAPTER 2. AUTOMATIC TEXT SUMMARIZATION IN THE LITERATURE

2.1 BACKGROUND

Summarization can be defined as the problem of highlighting the most salient information from the source (sources) text and transforming this information into final summary to be used by the user for its desired purpose [4]. This problem is not as straightforward as it may appear, since that different users have different interests, the information of importance to them would defer. Also as discussed above summaries provide different purposes and based on this purpose the important information would differ as well. In addition, there is the problem of evaluating the quality or goodness of the summary. In this chapter, we would look at some general information about different methods and architecture adapted to provide automatic summarization.

Summaries tend to differ at the most basic level in whether they are extracts or abstracts [6]. As mentioned before, extract summaries contain content as it appear from the source text while abstract is a paraphrasing of the important information in the original text. Currently, the majority of the approaches rely on providing extract summaries since they are easier to automate. Most process of any type goes through three phases of first analyzing the source text then determining the important parts and then synthesizing the output summary [4] [6]. The approach could be also "knowledge-poor" or "knowledge-rich" [6]. In knowledge poor approaches, they rely mainly on statistical and surface features of the text. Where a weighting method is adapted to provide a weight or a score for each sentence and extract the sentences with highest weights. Following is an illustration of sample architecture for extraction summarization without the need for domain knowledge.

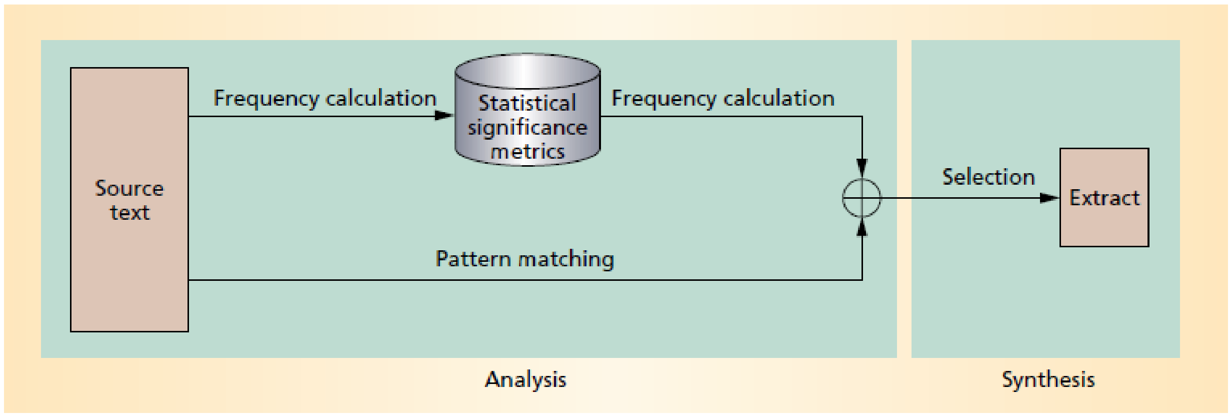


FIGURE 1 KNOLWEDGE POOR SUMMARIZATION ARCHITECTURE [9]

The analysis phase analyzes each sentence from the source text and gives it a score based on statistical significance metrics for example from term frequency counts or pattern-matching operation. Then in the synthesis process the top ranking sentences are extracted [6]. In the knowledge-rich approaches, information derived from related corpus is used in determining the importance of sentences. Figure 2 shows a sample architecture using knowledge-rich approach. In this architecture, a training data is used where each sentence from the training data goes through feature extraction providing a vector of features and labels from accompanying summaries is extracted. These feature vectors and labels are fed into a leaning algorithm that learns the classification rules for determining whether or not a sentence should be included into the summary. The test corpus then goes through the feature extraction and rule application to generate a summary based on the rule learned from the training corpus [6]. Although they are easier to implement, the problem with such approaches is that they extract sentences as is which could provide incoherent summaries. They also face problems like dangling anaphors. Anaphora takes its reference from another word or phrase so if the reference word of the sentence is not extracted in the final summary we are left with a dangling anaphora. There are efforts to address such problems like patching the output summary or not include sentences with anaphors [6].

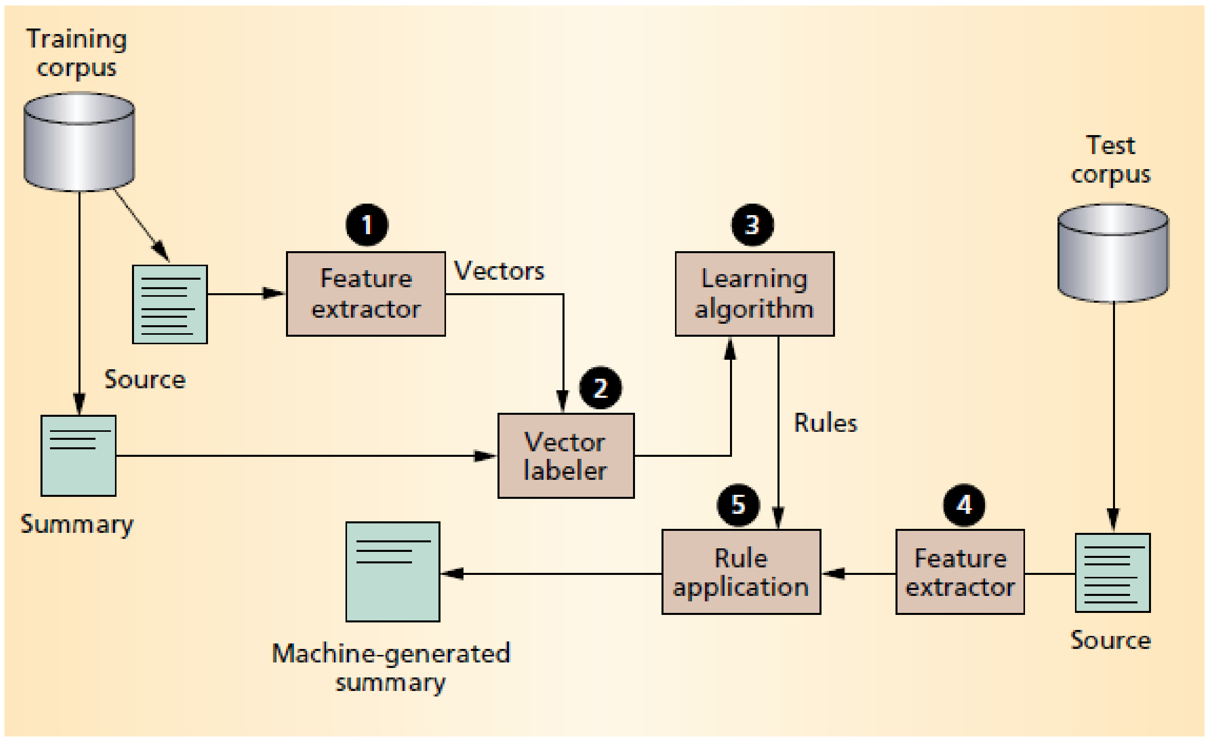


FIGURE 2 KNOLWEDGE RICH SUMMARIZATION ARCHITECTURE [10]

Other approaches make use of sophisticated natural language processing methods to provide better quality summaries. Figure 3 shows a sample architecture of such methods. Some of the approaches rely on traditional linguistic methods for parsing sentence syntactically along with using semantic information to annotate parse trees. Then the compacting procedures run on these trees to provide the summary. Other approaches follow the artificial intelligence route, trying to focus on natural language understanding. They parse the source text to provide conceptual representation of the entire content. The transformation phase in such approaches alters the representation through eliminating redundancy or irrelevant information or aggregate information providing a conceptual condensation of the original text. In the generation phase the text generator translates the structural or conceptual representation to produce a final coherent summary [6].

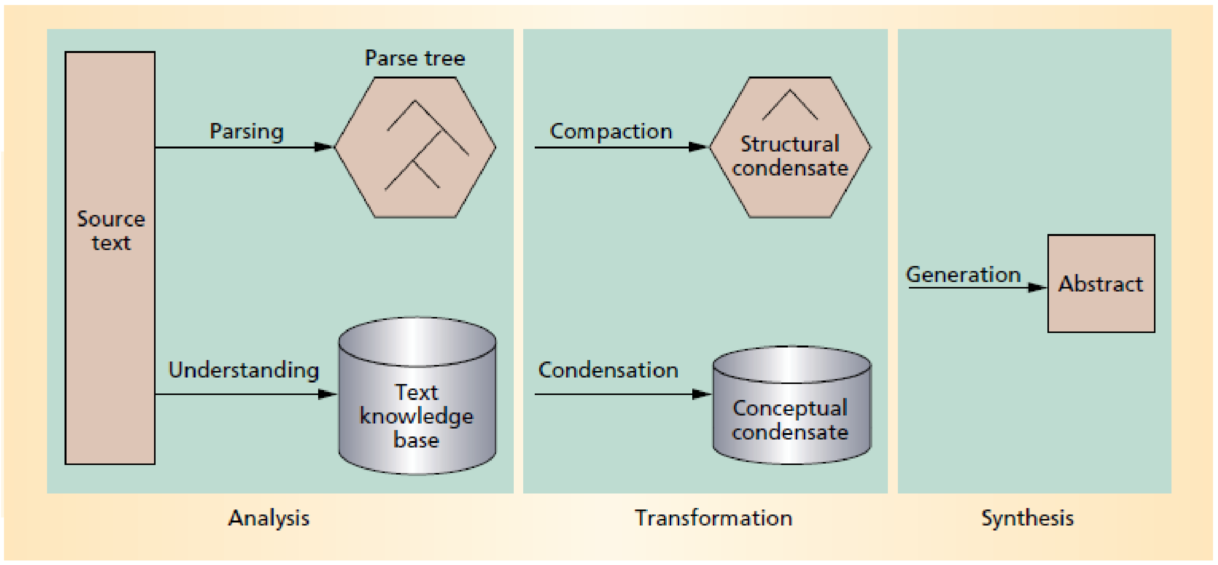


FIGURE 3 NATURAL LANGUAGE PROCESSING APPROACHES TO SUMMARIZATION [11]

These approaches provide more sophisticated summaries but are harder to implement. With recent work in corpus based NLP providing broad –coverage parser and comprehensive lexical resources such as Word-Net and Concept-Net along with more available corpus provide more support for such approaches to be adapted more in the problem of summarization [6]. In the literature review section we will take a look at various work done that utilizes these different approaches. Next we will look at different forms of summarization starting with document summarization and in the end investigate the issue of evaluating the quality of the output summary.

2.2 Document Summarization

2.2.1 Single Document Summarization

In single document summarization, the focus is to highlight the main important information and include it the summary. Most of the work done in the literature for single document summarization focuses on extractive methods. They use statistical and machine learning approaches to tackle the summarization problem. In this review, we will look first at the classical approaches that focused on statistical methods and then we will look at machine learning and deep natural language processing approaches along with approaches that make use of resources such as encyclopedia knowledge.

2.2.1.1. Classical Approaches

The early work in automatic text summarization focused mainly on summarizing technical papers. Reading through the literature most work would refer to Luhn work as one of the pioneering work in automatic summarization. Luhn work focused on creating automatic abstract to technical papers by the use of statistical methods. He devised a way to measure the significance of the content first for the words then for the sentences and determining which sentences to be included in the final auto-abstract. Stop-words were removed and similar spelling words were consolidated. First word frequency was counted, he based his hypothesis that writers normally repeat certain words for elaboration and proving their point which deems the word to be significant [12]. He sets an upper and lower bound for the frequency of the word to limit the effect of the repetition of common words. Then sentences significance is measured based on the position of significant word within the sentence. He called this the "significance factor" which reflects the number of occurrences of significant words within sentence and the linear distance between them due to the intervention of non-significant words [12]. By analyzing several documents, he came up with boundary for the distance between words to be considered in the computation of the factor and came up with limit of four to five non-significant words between significant words and dividing the number of significant word within this bracket of limit by the number of words to come up with a score. Sentences are then ranked according to this score and top ranking sentences are included in the abstract [12].

In [13] added methods for determining the significance of sentence are introduced. It uses the notion of “cue words” along with title and heading words and the sentence location to determine the significance of the sentence. They used corpus of 200 documents from fields of physical science, life science, information science and humanities. Human judges prepared target extracts based on specific criteria to be used in evaluating the resulting automatic extract. Each sentence was assigned a weight by four basic methods called Cue, Key, Title and Location. In the Cue method, the presence of pragmatic words such as “significant”, “impossible”, “hardly” gave sentence higher weight, the system used a corpus of bonus words that indicated positive relevance of words, stigma words that are negatively relevant and null words that are irrelevant. The Key method is similar to Luhn method discussed above which is based on the hypothesis that high-frequency content words are positively relevant. The title method gives positive weight for words in title and heading based on the hypothesis that words of the title are positively relevant. In the location method, words appearing early or very late in sentences of the document are hypothesized to be important since those sentences are considered as topic sentences. Thus those words are given higher weights. Also, words in sentences just below headings like “Introduction”, “Purpose” and “Conclusion” are given higher weights. Sentences weights were calculated based on these methods. It was found that the combination of Cue, Title and Location method provided the best results. Human judges were asked to rate the similarity between target extracts and the automatic extracts and they found a mean similarity rating of 66 percent. Also, statistical evaluation of the automatic extracts sentences as compared to the target ones found that 84% of sentences were extract-worthy [13].

These early works focused mainly on shallow surface methods to extract sentences for the summary. Nowadays with the advancement in natural language processing and machine learning methods, new approaches were introduced to tackle the problem of summarization. Following is a review of some of the recent work done to address such problem and the approaches used.

2.2.1.2. Summarization using Machine learning approaches

In [14], the authors tackle the summarization problem as a statistical classification problem. They use an already labeled dataset in order to develop a classification function that gives an estimate for the probability that the sentence should be used in the summary and then based on this probability sentences are ranked and top scoring sentences are selected in the summary. The features they used for the classification problem are sentence length cut-off, fixed-phrase feature, paragraph feature, thematic word feature and uppercase word feature. In the sentence length cut-off feature a threshold is given and each sentence above that threshold have this feature as true, this is based on the observation that short sentences tend not to be included in summaries. In the fixed-phrase feature, a set of predefined phrases are used and if sentence contain those indicator phrases or appears immediately after headings like "summary" or "conclusion" then the feature is true for that sentence. As for the paragraph feature they record information for the first ten paragraphs and last five paragraphs in the document and sentences are categorized as being paragraph-initial, paragraph-final and paragraph-medial. They use the most frequent words in document to define thematic word feature and top scoring sentences are computed and then if sentence is among these sentences then its feature is true for thematic word feature. Similar method is used to compute the uppercase word feature based on the notion that proper names are often important and represented as uppercase word. For each sentence they compute the probability of it being included in the summary based on the features above using Baye's rule as follows [14]

$$P(s \in S \mid F_1, F_2, \ldots F_k) = \frac{\prod_{i=1}^{k} P(F_i \mid s \in S) . P(s \in S)}{\prod_{i=1}^{k} P(F_i)}$$

Their training set consisted of 188 document/summary pairs from 21 publications in scientific and technical journals. Some documents had no accompanying summary so they employed professional abstractors to produce an abstract. Evaluating their approach they found that the summarizer replicated 35% of the information in the manual summaries they also found that the combination of paragraph, fixed phrases and length cut-off features produced best results [14].

In [15], the authors also used classification methods to tackle the summarization problem and applied it to Korean texts. They classify sentences into different thematic categories and apply a classifier based on individual features and combine the result from all features using Dempster-Shafer combination rule. In addition to the cue words, keywords and position features, they experimented with centrality, title resemblance and text component features [15]. The centrality feature measures the similarity between a sentence and the rest of the document while the title resemblance features measures how similar a sentence is to the title of the document. The text component feature divides the document into several parts and is used as a filter to keep only sentence belonging to major parts of the document. The system consists of two processes the training process and the summary generation process. The first phase extracts the necessary statistical information from the corpus and the second one uses the information from the first phase to calculate the likelihood of a sentence belonging in a summary. They used 30 documents in the test set, using human judges to select sentences from the summary they compared it to the system generated and got a 67% recall and 40% precision. Their work introduced the text component identification where upon examination they found that in technical documents the sentences of summary belong to one of the components of background or main theme or explanation of the document structure or future work sections in the document. They manually tagged the sentences belonging to each category and analyzed it to extract lexical clues for the sentences belonging to each of such category, they used this method to remove noise from the full document [15].

After 2002, the Document Understanding Conference (DUC) dropped the task of summarization of single news articles since that no system could outperform the baseline method using statistical significance [16]. However, the work done by [16] shows that using neural nets ranking algorithms and third-party datasets to enhance the sentence features can outperform the baseline method. The authors present NetSum which is a new approach to automatic summarization using neural net. They apply features based on news search query logs and Wikipedia entities. They use the RankNet learning algorithm to train pair-based sentence ranker to give every sentence in the document a score based on which important sentences are extracted in the summary. RankNet is a pair-based neural network algorithm that rank a set of inputs which consists of sentences in the given document. The training dataset is based on pairs of sentences (S_i,S_j) where S_i should be ranked higher than S_j, they use the ROUGE score to determine which sentence is ranked higher. ROUGE score is a score given by ROUGE evaluation system to evaluate the generated output summary as opposed to ideal human summaries. The cost function is the probabilistic cross-entropy cost function and uses a modified version of the back propagation algorithm for two layer nets based on optimizing the cost function by gradient descent [16]. It takes as input set of samples containing label which is ROUGE score and set of features, their feature list is as follows

TABLE 1 FEATURES USED IN NETSUM [16]

Symbol	Feature Name
F(Si)	Is First Sentence
Pos(Si)	Sentence Position
SB(Si)	SumBasic Score
SBb(Si)	SumBasic Bigram Score
Sim(Si)	Title Similarity Score
NT(Si)	Average News Query Term Score
NT+(Si)	News Query Term Sum Score
NTr(Si)	Relative News Query Term Score
WE(Si)	Average Wikipedia Entity Score
WE+(Si)	Wikipedia Entity Sum Score

They use third party sources as news query and Wikipedia entity. The news query is retrieved from Microsoft's news search engine. Their hypothesis was that a sentence with higher number of news query terms should be a better candidate highlight [16]. They report that for 73.26% of document their system produces a summary with ROUGE-1 score that is equal to or better than the baseline and for 24.69% of document the score is equal to the baseline. For ROUGE-2 score their system performs equal to or better than baseline on 73.19% of documents and equal to the baseline on 40.51% of documents [16].

2.2.1.3. Summarization using Encyclopedia Knowledge

In [3], a single document summarization using Wikipedia is presented. The authors present their approach to be a proxy based approach to process document enroute to mobile devices and send only summary to the device. They first map each sentence to Wikipedia concept by using Lucene [1] engine to retrieve hits to Wikipedia articles. Lucene engine is a high-performance, full-featured

[1] Lucene http://lucene.apache.org/core/

text search engine library written entirely in Java. The entire Wikipedia corpus is indexed using Lucene engine and each sentence in source text is input as a query to the engine. The retrieved Wikipedia articles titles are extracted and mapped to sentence as concepts. Then a matrix MXN where M is total number of sentences and N is the number of concepts in given document is constructed to represent a bipartite graph of the concept mapping. One set of nodes in the graph represented the sentence and other set represent concepts and the edge between nodes in the first set and second indicate a mapping between sentence and concept. The matrix represents these connections and is used in the summarization algorithm where sentences in document that got mapped to the concepts with largest number of hits will be selected as output in the summary. The user selects minimum and maximum threshold for concepts and concept within this range are used to select sentences. They evaluated their system using the DUC 2002 single document summarization task and using ROUGE package. They reported the below results for ROUGE-1 average recall numbers [3].

TABLE 2 SUMMARIZING ROUGE RECALL RESULTS IN [3]

Wikipedia concepts threshold	ROUGE recall
2	0.468
3	0.4586
MAX	0.4368

This algorithm have limitation where a sentence can be selected for being mapped to only one concept, having more concepts mapped to the sentence makes it more relevant. Also the summary size is not controlled and it might result in large summaries [3].

2.2.1.4. Summarization using Deep Natural Language Processing Approaches

The following approaches use more sophisticated natural language processing approaches to tackle the summarization problem.

2.2.1.4.1. Summarization using Lexical Chains

In [17], authors use lexical chains as a mean to represent a text rich enough for good quality indicative summaries and easy to extract. Lexical chains provide a way to represent the lexical cohesive structure of the text and have been used in fields like information retrieval. Cohesion of text is the way text “stick together” to provide one meaning. Lexical cohesion arises from the semantic relationships between words [18] where relation between words is recognized. In their approach the authors propose a method to develop lexical chain that takes into account the different alternatives of word sense and choose the best alternative. The approach first processes the source creating lexical chains where they define a component which is a list of interpretations of words that are exclusive of each other [17]. They use WordNet to provide different senses for the word. Figure 4 shows the different component created to represent the different alternative for senses of words in the phrase “Mr Kennedy is the person that invented an anesthetic machine which uses micro-computers to control the rate at which an anesthetic is pumped into the blood…”.

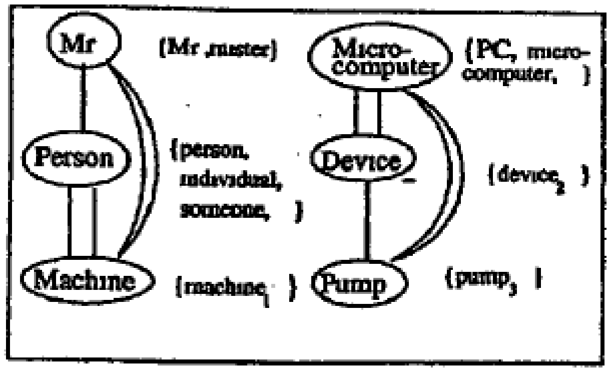


FIGURE 4 USING LEXICAL CHAIN TO REPRESENT TEXT [19]

The component represents the different senses for word “machine” and “person” and the relation between these senses. The component with the most connections is considered the best representation and the more cohesive. Then a score of chain is computed according to number and weight of the relation between chain members. And the score of the interpretation will be sum of the score of its chains. So their algorithm processes the text developing all possible interpretations and prunes the weakest ones when the interpretations are larger than a certain threshold and at the end selects from each component the strongest interpretation. The author claims that given a good measure of strength, the chain with the strong score would be a better indication of the central topic of the text rather than the use of frequent words [17]. Using lexical chain, words with related concepts will occur in the same chain and so if a concept is represented by different words appearing with low frequency it will appear that this concept is a strong one having a chain with many connections rather than using word frequency which won’t be able to capture this concept. They also distinguish noun components as candidate words using words like “sea level” to represent one noun. The authors collected a data set of 30 texts extracted from popular magazines like “The Economist” and “Scientific American”. They then manually ranked chains in terms of relevance to the main topics and computed different formal measures of the change including chain length, distribution in the text and the text span covered by the chain. From their results they found that the length and the homogeneity index where good predictors of the strength of the chains. They gave the following definition for length and homogeneity index [17]:

Length = The number of occurrences of members of the chain

Homogeneity index = 1- the number of distinct occurrences divided by the length

And they defined the score of the chain to be as follows [17]:

Score (Chain) = Length * Homogeneity

They evaluated that the strong chains are the ones that satisfy their strength criteria which they defined as [17]:

Score(Chain) > Average(Scores) + 2 * StandardDeviation(Scores)

From their experiments they found that the average number of strong chains selected by this method was 5 for texts of 1055 words from original 32 chains.

After identifying the strong chains the next step in the approach is to extract the full sentences for the summary output. They call them significant sentences. They tried three different heuristics to extract these sentences. The first heuristic they choose the sentence where the first appearance of a chain member in the text occurs. The problem with this approach is some words are more representatives of the topic than other. Thus they defined criteria by which a chain member is determined whether it is appropriate representative of the chain based on its frequency of occurrence in the chain. They called these words “representative words” and they have frequency in chain no less than the average word frequency in the chain. In the second heuristic they choose the sentence in which a representative chain member appears first in the text. They found that this heuristic gave almost same results as the first. In the third heuristic they try to identify the text unit where the global topic of the text is its focus and extract sentences related to the topic from this segment. They characterize that text unit as “a cluster of successive segments with high density of chain members” [17]. So for each chain, the find the text unit where the chain is highly concentrated then extract the sentence with the first chain appearance in this central unit. They compute the concentration by the number of chain member occurrences in a segment divided by the number of nouns in the segment where a chain will have a high concentration if its concentration is the maximum of all chains. In each of these heuristic on sentence is extracted per chain. They found out that the first two heuristics yield almost same results however when they produce different results the second heuristics' results are better. They found that despite the fact that the third heuristic appears to be more sophisticated it gave the least indicative results [17].

This work still needs to address issues like dangling anaphors and they don't provide any way to control the length and level of details in the summary [17].

2.2.1.4.2. Summarization based on rhetoric structure theory

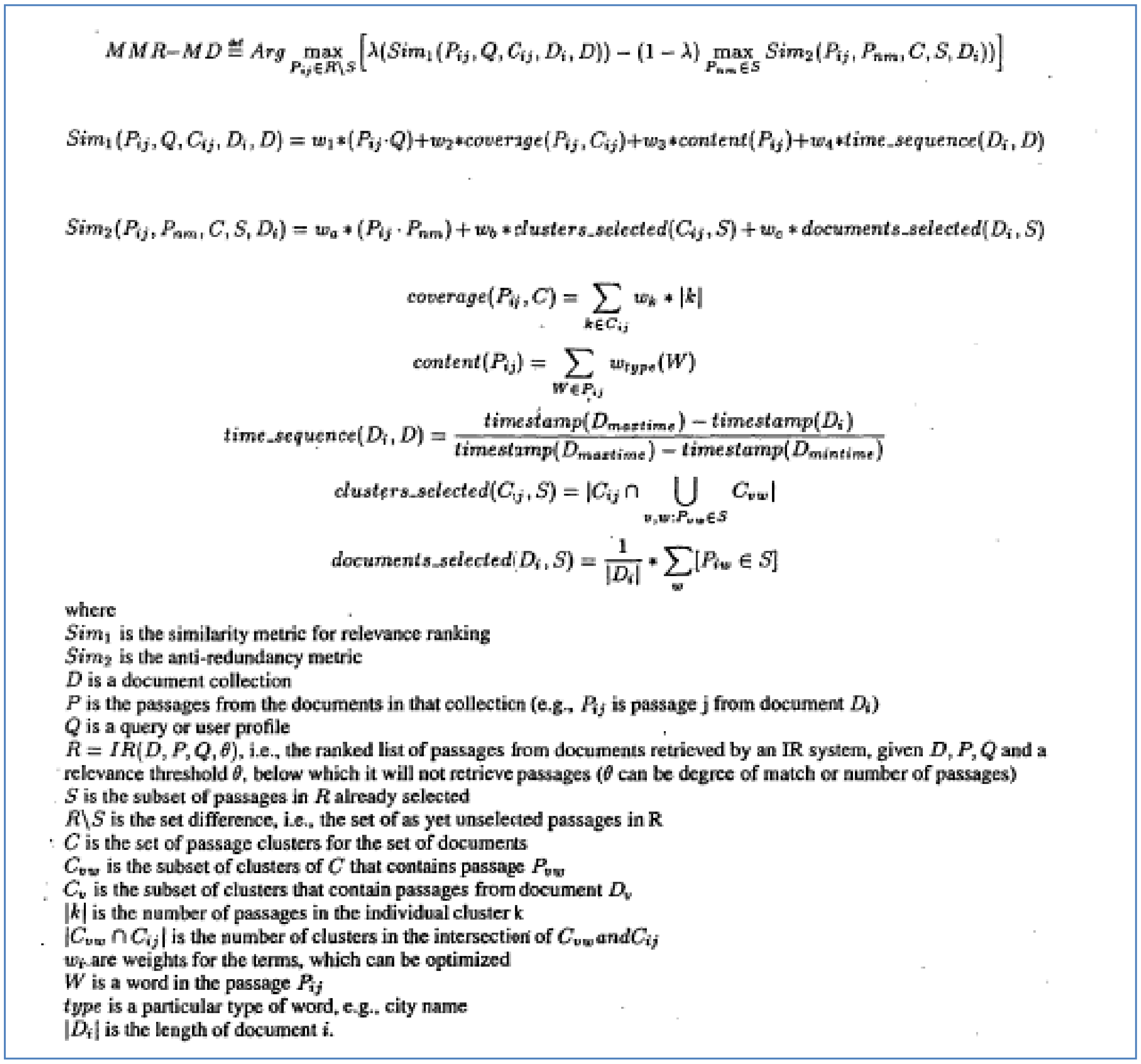
Rhetoric structure theory (RST) is one of the popular discourse theories that uses the notion of rhetoric relation between two non-overlapping text spans called nucleus and satellite [20]. Nuclei present what is more important from the writer point of view than the satellite and the nucleus of a rhetorical relation is comprehensible without the satellite however satellite can't be comprehended without nucleus. Rhetoric relation could be represented in rhetorical structure trees. In [21], the authors developed a computational model of discourse for Japanese expository writing and used it to generate an automatic abstract. They used two layers for rhetoric structure, first is intra-paragraph one between sentences and the second is inter-paragraph one between paragraphs. They focused on connective expressions as an inter-sentence relationship that clarifies the principle of arguments. They divided the connective expressions of grammatical connectives and sentences predicates into thirty four categories with examples like reason, example, extension and rephrase [21].A series of steps was used to extract a binary tree representing rhetorical structure tree, these steps are sentence analysis, rhetorical relation extraction, segmentation, candidate generation and preference judgment [22]. Trees were pruned to reduce the sentence while keeping important parts. Evaluation was with respect to sentence coverage using a dataset of 30 editorial articles of Japanese newspaper. Upon evaluation of sentence coverage it was about 51% with most important key sentence coverage of 74% [22].

2.2.2. Multiple Documents Summarization

Multiple documents summarization poses more challenges than single document summarization. The compression ratio, speed and redundancy become more critical when we try to summarize a set of multiple documents. When facing multiple documents they could be of dis-similar information thus the system should first cluster similar document and provide summary for each collection of document, or the document could be topically related and related to a user-specific query. One of the most important issues is that multiple document summarizer -especially in topically related document -should have anti-redundancy methods so that information common between documents is included only once. Also there is the issue of temporal dimension where later information may become more important and override earlier information. The compression ratio in hundred of related document will be much smaller than a single document and poses more challenges, in addition to the co-reference problem [23]. There could be different types of multiple document summarizers , for example they could provide summary for the common section of the document set or adding to it unique section from each document. Some systems provide summary for the centroid document of the cluster only or adding to it the outliers documents as well. Following is a review of two of the main approaches in the literature for multiple document summarization systems.

2.2.2.1. Multi-Document Summarization by Sentence Extraction (Maximal Marginal Relevance Multi-Document (MMR-MD))

In [23], the authors build on single-document summarization methods by using the information available about the document set and the relationship between the documents in the set to extract text for the summary. They primarily focus on using only domain independent techniques focusing on statistical processing and dealing with reducing redundancy while maintaining the important information, they also provide parameterized modules so that system could be adapted for different genres characteristics. They focus on maximizing marginal relevance of the passage being selected in the final summary. A passage has a high marginal relevance if it is relevant to the user query and useful to the summary while having minimal overlapping between already selected passages [23]. They devised the metrics shown in Figure 5 [23] for measuring the marginal relevance of a passage. In Sim1, the first term measures the cosine similarity between the query and the document. Second term measures the coverage score for the passage by whether it is in one or more clusters and the size of the cluster. The third term reflects the information content of the passage and the final term is used to indicate the temporal sequence of the document in the collection. This allow for more recent information to have higher weight. In Sim2, the first term uses the cosine similarity to compute the similarity between the passage and the already selected passages. The second term penalizes passages that appear in clusters of already selected passages and the third term penalizes document from which passages are already selected which is inversely proportional to the document length to allow longer document to have more passages included. Thus Sim1 measures the relevance of the passage to the query and Sim2 is for measuring the novelty of the passage. Their system works as follows, it first segments the documents into passages and indexes them. Then they identify the passages relevant to the query using cosine-similarity. Then they apply the MMR-MD metric to score and select passages and users can select the number of passages of the summary. Finally they reassemble the selected passages into a summary document. They used the TIPSTER provided corpus of Associated Press and Wall Street Journal. They used set of 200 documents and used the topic provided as query. On average these document where 31 sentences long and have a total of 6115 sentence. When they generated a summary of 10 sentences they had a sentence compression ratio of 0.2% and character compression of 0.3% [23].



$$MMR\text{-}MD \stackrel{\text{def}}{=} Arg \max_{P_{ij} \in R \setminus S} \Big[\lambda(Sim_1(P_{ij}, Q, C_{ij}, D_i, D)) - (1-\lambda) \max_{P_{nm} \in S} Sim_2(P_{ij}, P_{nm}, C, S, D_i))\Big]$$

$$Sim_1(P_{ij}, Q, C_{ij}, D_i, D) = w_1 * (P_{ij} \cdot Q) + w_2 * coverage(P_{ij}, C_{ij}) + w_3 * content(P_{ij}) + w_4 * time\_sequence(D_i, D)$$

$$Sim_2(P_{ij}, P_{nm}, C, S, D_i) = w_a * (P_{ij} \cdot P_{nm}) + w_b * clusters\_selected(C_{ij}, S) + w_c * documents\_selected(D_i, S)$$

$$coverage(P_{ij}, C) = \sum_{k \in C_{ij}} w_k * |k|$$

$$content(P_{ij}) = \sum_{W \in P_{ij}} w_{type}(W)$$

$$time\_sequence(D_i, D) = \frac{timestamp(D_{maxtime}) - timestamp(D_i)}{timestamp(D_{maxtime}) - timestamp(D_{mintime})}$$

$$clusters\_selected(C_{ij}, S) = |C_{ij} \cap \bigcup_{v,w: P_{vw} \in S} C_{vw}|$$

$$documents\_selected(D_i, S) = \frac{1}{|D_i|} * \sum_{w} [P_{iw} \in S]$$

where
Sim_1 is the similarity metric for relevance ranking
Sim_2 is the anti-redundancy metric
D is a document collection
P is the passages from the documents in that collection (e.g., P_{ij} is passage j from document D_i)
Q is a query or user profile
$R = IR(D, P, Q, \theta)$, i.e., the ranked list of passages from documents retrieved by an IR system, given D, P, Q and a relevance threshold θ, below which it will not retrieve passages (θ can be degree of match or number of passages)
S is the subset of passages in R already selected
$R \setminus S$ is the set difference, i.e., the set of as yet unselected passages in R
C is the set of passage clusters for the set of documents
C_{vw} is the subset of clusters of C that contains passage P_{vw}
C_v is the subset of clusters that contain passages from document D_v
$|k|$ is the number of passages in the individual cluster k
$|C_{vw} \cap C_{ij}|$ is the number of clusters in the intersection of $C_{vw} and C_{ij}$
w_i are weights for the terms, which can be optimized
W is a word in the passage P_{ij}
$type$ is a particular type of word, e.g., city name
$|D_i|$ is the length of document i.

FIGURE 5 METRICS FOR MEASURING MARGINAL RELEVANCE OF PASSAGE [24]

2.2.2.2. Centroid-based summarization of Multiple document (MEAD)

In [8], the authors present a multi-document summarizer called MEAD that summarizes multiple documents based on cluster centroids produced by topic detection and tracking system. The authors built the summarizer on their topic detection and tracking system called CIDR which identifies all articles on an emerging event using modified TF*IDF. TF*IDF stands for the product of the term frequency and its inverse frequency in the whole corpus. Then they produce clusters of new articles on the same event. MEAD takes the clusters produced by CIDR as input and uses it to identify sentences that are central to the topic of the cluster. MEAD uses cluster centroids that consist of words that are central to the articles in the cluster. Compared to Maximal marginal relevance mentioned above, MEAD is designed for queried independent summaries [8]. They experimented with a small corpus of 27 document organized in 6 clusters. A centroid produced from CIDR is a pseudo-document consisting of words which have Count*IDF above a pre-defined threshold where count is the average number of occurrence of the word across the entire cluster. They hypothesized that sentences which contain words from the centroid are more indicative of the topic of the cluster. The algorithm works as follows; it is given a cluster of articles and value of compression rate. For each sentence in the cluster it gives it a score based on a set of parameters. The parameters are centroid score, position and overlap between first sentence. It lays them in the same order as they appear in original document with document ordered chronologically. Score is calculated as follows [8]

$$SCORE(s) = \sum_i (w_c C_i + w_p P_i + w_f F_i)$$

$$where\ i(1 \leq i \leq n) is\ the\ sentence\ number\ within\ the\ cluster$$
$$C_i \text{ is the centroid score}, P_i\ is\ the\ position\ of\ the\ sentence\ and$$
$$F_i\ is\ the\ overalp\ between\ the\ sentence\ and\ the\ first\ sentence$$
$$w_c, w_p\ and\ w_f\ are\ the\ weights\ given\ for\ C_i, P_i, F_i\ respectivaly$$

The centroid score (C_i) measures the centrality of a sentence to the overall topic of a cluster by comparing it to the centroid's words. The position of the sentence (P_i) decreases linearly as the sentence gets further away from the beginning of the document. The overlap between the sentence and the first sentence (F_i) is the inner product of the TF*IDF weighted vector representation of the sentence and that of the first sentence (or title) of the document.

For evaluation they used the Lead-based method in which the positionally first (n*r/c) sentences from each cluster are picked as baseline for evaluation where c is the number of clusters until length of summary is reached. Table 3 shows the normalized performance of MEAD for six clusters at nine compression rate. It performed better than Lead-based method in 29 out of 54 cases where in the largest cluster it outperformed it in all compression rates.

TABLE 3 NORMALIZED PERFORMANCE OF MEAD [8]

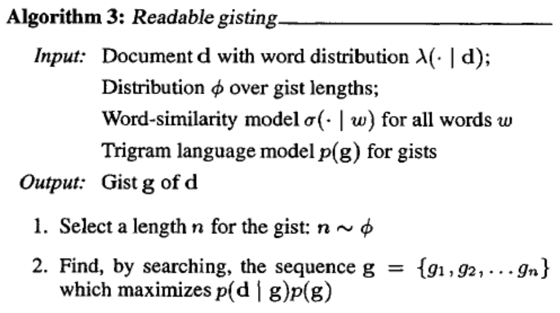
Clusters	**10%**	**20%**	**30%**	**40%**	**50%**	**60%**	**70%**	**80%**	**90%**
Cluster A	0.855	0.572	0.427	0.759	0.862	0.910	0.554	1.001	0.584
Cluster B	0.365	0.402	0.690	0.714	0.867	0.640	0.845	0.713	1.317
Cluster C	0.753	0.938	0.841	1.029	0.751	0.819	0.595	0.611	0.683
Cluster D	0.739	0.764	0.683	0.723	0.614	0.568	0.668	0.719	1.100
Cluster E	1.083	0.937	0.581	0.373	0.438	0.369	0.429	0.487	0.261
Cluster F	1.064	0.893	0.928	1.00	0.732	0.805	0.910	0.689	0.199

2.3 WEB PAGES SUMMARIZATION

Summarization of web pages faces added difficulties than document summarization, they lack the coherent discourse of text [25], content is provided within HTML tags that include images, advertisement and other content that is not related to the main topic of the page. Sentence and text units are harder to identify and isolate from all the tags and other jumble of text available in HTML page. New method should be researched to segment text and create summaries for web pages. Also the use of hyperlinks and click-through data provide resources for summarizing web pages in relation to its links to other web pages. Following is a review of some of the work done so far to tackle the problem of web page summarization. The first approach makes use of content information in the web page to try and synthesize a web page summary. The next approach utilizes the clickthrough data gathered from a search engine to enhance the method of summarization. The last approach leverages the context information from the web by investigating the structure of hypertext and how people describe information in it to provide a way for web page summarization.

2.3.1 Web Page "Gisting"

OCELOT system tackles the problem of web page summarization by trying to synthesize a summary rather than trying to extract one [25]. They focus on synthesizing the "gist" of the web page. It relies on statistical models to guide the choice of words and their arrangement in a summary. Their system goes through content selection to determine which words should be included in the summary and then word ordering and arranging selected words into a summary then for a given web page they search the sequence of words which is optimal in the dual sense of content and readability [25]. They propose two methods for content selection the simplest of which is the selection of words according to the frequency of appearance. Another approach is to allow for words that don't appear in page to appear in the gist. They use the n-gram mode of language in order to order the appearance of words in gist. They apply both the content and ordering simultaneously by using the generic Viterbi search techniques to efficiently find near-optimal summary where they search over all candidates of gists to find the one that maximizes the product of content selection term and surface realization term [25]. They propose three models for gisting. First approach is the "bag of words" approach where they start by selecting a length n for the summary according to some probability distribution over possible lengths. Then for each of the assigned positions in the gist they draw a word at random from the document and fill the current slot. This model makes the assumption that the more frequently a word appears in a document the more likely it will be included in the gist. The second approach accounts for unseen words since the first approach only includes words appearing in page. The idea is to draw a word like the previous approach and then replace it with a related word say a synonym and then adding it to the gist, they determine which word to use in substitution based on probability distribution where the value will be high for closely related words. They call this algorithm the expanded-lexicon gisting [25]. The third approach focuses on generating readable summary. They expand on the second approach by enforcing that the sequence of words comprising a candidate gist are coherent [25]. They use the a priori probability of seeing that string of words in text. They use a trigram model for that. Following is the algorithm they use for the third approach [25].



```
Algorithm 3: Readable gisting
  Input:  Document $\mathbf{d}$ with word distribution $\lambda(\cdot \mid \mathbf{d})$;
          Distribution $\phi$ over gist lengths;
          Word-similarity model $\sigma(\cdot \mid w)$ for all words $w$
          Trigram language model $p(\mathbf{g})$ for gists
  Output: Gist $\mathbf{g}$ of $\mathbf{d}$
  1. Select a length $n$ for the gist: $n \sim \phi$
  2. Find, by searching, the sequence $\mathbf{g} = \{g_1, g_2, \ldots g_n\}$
     which maximizes $p(\mathbf{d} \mid \mathbf{g})p(\mathbf{g})$
```

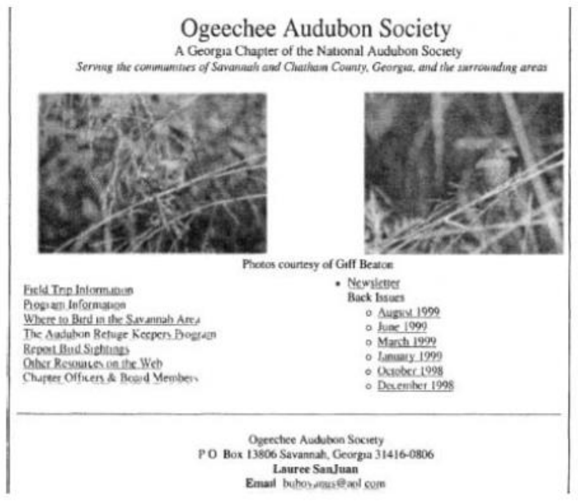
FIGURE 6 OCELOT ALGORITHM FOR READABILITY GISTING OF WEB PAGES [26]

They use the corpus from the Open Directory Project which is a directory of web pages that relies on a large number of volunteers to grow and maintain it where for each web site there is a summarized version done by one of the volunteers [25]. They processed the text in the web pages by normalizing text removing stopwords and converting it to lowercases. They removed links, images and meta-information from the page. They removed pages with adult oriented content, removed HTML markup and excluded pages with frames as well as duplicated pages. Then they partitioned the set to training set and test set. They were left with 103,064 pages in the training set and 1046 in test set. After processing the average length of the summary is 13.6 words and that of the page is 211.1 words. They train a statistical model for gisting similar to the ones used in machine translation where the "two languages" are the "verbose langue of documents and the succinct language of gists" [25]. They use alignment between sequences of words which captures how words in gists produce the words in a web page to estimate parameter in the model. They use maximum likelihood estimation in particular the expectation-maximization (EM) algorithm to compute the parameters.

Figure 7 is an example of their OCELOT gist for a sample web page. For the first approach, they defined word overlap as a method of evaluation, which is the number of words that appeared in both the hypothesized and actual gist divided by the size of the hypothesized gists. The following table summarizes the results [25]:

TABLE 4 OCELOT ALGORITHM 1 WORD OVERLAP EVALUATION [25]

Summary length	Word overlap
4	0.4116
5	0.3489
6	0.3029
7	0.2745
8	0.2544
9	0.2353



Ogeechee Audubon Society

A Georgia Chapter of the National Audubon Society

Serving the communities of Savannah and Chatham County, Georgia, and the surrounding areas

Photos courtesy of Giff Beaton

Field Trip Information
Program Information
Where to Bird in the Savannah Area
The Audubon Refuge Keepers Program
Report Bird Sightings
Other Resources on the Web
Chapter Officers & Board Members

- Newsletter
  Back Issues
  - August 1999
  - June 1999
  - March 1999
  - January 1999
  - October 1998
  - December 1998

Ogeechee Audubon Society
P O Box 13806 Savannah, Georgia 31416-0806
Lauree SanJuan
Email buboyamus@aol com

Open Directory gist: a chapter of the national audubon society serving the communities of savannah chatham county and the surrounding areas

OCELOT ***gist:*** audubon society atlanta area savannah georgia chatham and local birding savannah keepers chapter of the audubon georgia and leasing

FIGURE 7 EXAMPLE OF OCELOT GIST FOR A WEB PAGE [27]

They evaluated the language model in isolation by calculating the probability which the language model assigns to a set of unseen Open Directory gists where the higher the probability the better the model. They used the perplexity measure that could be thought of as the average number of guesses the language model must make to identify the next word in a string of text comprising a gist drawn from the test set. For the test collection the perplexity of the language model with 362 as opposed to weaker bigram and unigram models whose perplexity is 536 and 2185 [25].

2.3.2 WEB PAGE SUMMARIZATION USING CLICK-THROUGH DATA

In [28], the author uses the click-through data obtained from search engines in order to enhance web page summarization task. When a user enters a query in search engine, a list of web pages is retrieved and the user clicks on pages of interest. The server side accumulates the collection of click-through data represented in form of triples <u,q,p> where u is the user and q is the query and p is the clicked page. Thus such data records how Web users find information through queries. Such collection of queries related to one web page could represent how the pages are related to issued queries and thus could reflect topics presented in the page. The authors conducted an experiment to investigate whether the query word set retrieved from click-through data is related to the web page topic. They use pages crawled from the open directory project and clickthough data from MSN search engine. They asked human evaluators to conduct manual summarization on a set of pages. They found that about 71.3% of sentences in the created summaries contained words from the query set as opposed to 58% of sentences in the original web pages. Thus they concluded that using the query set words in summarization should improve the quality of its summarization. They adapted two methods from the literature for summarization, one is the adapted significant word method (ASW) and the other is the adapted latent semantic analysis method (ALSA). The adapted significant word method based on Luhn's algorithm tackled in document summarization section above. They modified the approach so that they use both the local contents of web page and query terms collected from the clickthrough data to decide whether a word is significant or not. The significant factor of word is measured in weighted combination between its frequency in the local text and its frequency in the query set. They rank all the words and pick the top N% as significant words and then apply Luhn's algorithm to compute the significance factor for each sentence. In the adapted latent semantic analysis method they build on the method proposed in [29] that uses the Latent semantic analysis method to capture the latent relations between terms. Where documents are represented by column vectors and each vector represent a latent concept with a value representing its importance. Concept importance is measured and then sentences with the largest importance factor are selected. They proposed a variation of this method where if a term occurs in the query its weight is increased to its frequency in the query set. They aim to extract sentences whose topics are related to the ones reflected in the query words. In both of these approaches they wanted to utilize the knowledge of query terms to select significant words since that some words frequency in the actual page is low however they tend to be significant and appear in the query, adapting the method in this way will allow for such words to be more significant. They also propose a method for pages that are not covered in the click-through data through building a hierarchical lexicon using the click-through data and apply it to help summarize such pages. All open directory project pages have been manually organized into a hierarchical taxonomy, the authors combine such knowledge with one extracted from click-through data to build a thematic lexicon. The lexicon contains all query terms submitted to browse each category and weights for these terms which measures the likelihood that Web users will use such terms to locate a page in such category. They build such lexicon as follows for each category they investigate pages in it covered by the click-through data adding its query words to the lexicon of such category and its parent category. And the query frequency is added to its original weight. If a page belongs to more than one category its query terms are added to all lexicon associated with each category. Then the term weight is multiplied by its inverse category frequency which is the reciprocal of its frequency occurring in different categories of the hierarchical taxonomy. After building the lexicon they use it to summarize pages not covered in the click-through data. For such page, the lexicon associated with its category is looked up and they use the method proposed above where the weight in the lexicon is used to select significant words or updated the term-sentence matrix [28]. Their data set consists of two different sets, the first one consists of 90 pages and three human evaluators were employed to summarize those pages through extracting sentences that they see most important. The other data consists of 260,763 pages with click-through data, they extracted their content data and their description from the metadata to be considered the ideal summary. They used precision, recall and F1 measure for performance measure as well as ROUGE method. They reported improvement from the original methods on first dataset with 20.7% improvement for the ASW method and 12.9% for the ALSA method when measure by precision. They also reported improvement of 11.5% for both methods when measured using ROUGE method. For second dataset they used only ROUGE-1 measure which should show improvement for the web page summarization using the click-through data [28].

2.3.3 Web Page Summarization using hypertext structure

In [30], the authors investigate a new way to provide web page summarization by using the structure of the hypertext and the way people describe information in it. They built their system InCommonSense to provide web page snippets to demonstrate their approach. This system takes advantage of the paragraph convention found in Web hypertext. It extracts annotations, notes and descriptions that people write about other web pages. Upon investigating over 250,000 web pages, they found that there is a pattern in the way people write and annotate in hypertext and different patterns of linking within the limits of a paragraph. They found that the pattern where a paragraph begins with an anchor followed by text is useful for predicting the topic of the linked document. There system starts by taking a web document and looks for all other documents linking to it. They look for the previously described pattern for linking to it as shown in Figure 8 below.

It collects information through querying search engines for document linking to the required document. Then it analyzes those pages looking for paragraphs markup cues, it looks for segments of text that have empty spaces before and after the text and segments of text that are marked as different entities.

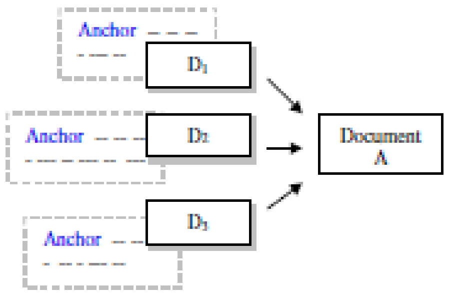


FIGURE 8 INCOMMONSENSE DETECTING RELATIONS BETWEEN WEB DOCUMENTS [31]

It allows for a determiner like The, This or A to proceed the anchor as well. It currently processes up to 220 documents relating to a single URL in one run using search engines like Google, HotBot, AltaVista and Infoseek. They conducted an experiment to determine the descriptive value of the snippets collected with InCommonSense in order to build a filter for identifying good description in the data. They used the results to determine which language or textual features to use to automatically predict the good and bad descriptions and use them to filter the description. Subjects were presented with a web page and a collection of description as collected from the web. They were asked to read the page and assigned each snippet value from 1 to 5 where 1 being bad and 5 being good. Overall all, they rated 252 snippets with the participation of 746 different subjects. For each snippet they calculated mean score and confidence interval. They sorted them into good snippets, bad snippets and mixed scored snippets. If the mean plus/minus the interval stayed above the value of 3 then it is considered good while if it stayed below 3 it was bad, if it crossed the 3 threshold it was considered mixed. They analyzed the data to select features to filter through the description. They used an off-the-shelf machine learning tool (See5) to train the classification tool. They used features to account for length, punctuation, use of personal pronoun, use of acronyms, use of terms indicating content like 'about', position of punctuations, position of verbs, text beginning with capital letter and term repetition ratio. Using See5, 16 rules were hard coded in the system to create a fast and independent description filter. They evaluated their approach to commonly used one by search engines where the top X words are taken from the document which they called the AltaVista style and when the query terms are highlighted and the surrounding context is taken which they called the Google style. They conducted an experiment where participants were assigned a task composed of a short information need description and a reason for choosing the query. They were asked to read the task and the assigned query and a set of results was displayed with either of the three styles (AltaVista, Google, InCommonSense) [30]. They had to choose the one, most appropriate result. They then viewed the result they choose and answer four questions rating them on a 7 point scale. The questions are : are you happy with the result, it was easy to find the information I need, I read the textual snippet given with the results in order to make a decision and I usually read the textual snippets given for search results. 738 people participated in the experiment. Analyzing the results, they found that people are happy with their results regardless of the display. They observed that they spent more time reading snippet from InCommonSense. The main finding is that for the second question of how easy is it to make a decision there was a difference between people who interacted with InCommonSense and other styles. They found that in terms of ease of interaction, the textual output of InCommonSense is superior to the output of the other two styles [30].

2.4 Other Forms of Summarization

There exists other form of summarization in the literature, below is a review of some of those work. We review a work done on multilingual summarization using Wikipedia. We also look at opinion summarization that makes use of highly redundant information to provide a form of shallow abstract summarization with an application on people’s review, we also look at sentence compression as done through graph representation of text approach. Both of these works influence our approach as explained in the approach section above. We finish with a look at a different form of summarization that aim at providing image captioning by investigating the content of web pages with content related to the image place or location.

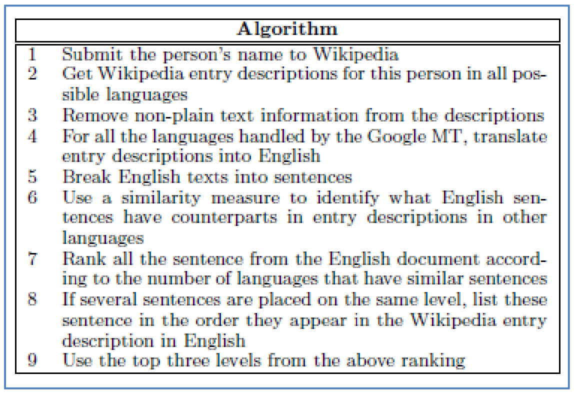
2.4.1 MULTILINGUAL WIKIPEDIA SUMMARIZATION

Wikipedia entries provide information about various categories like people, events, countries..etc in many languages. These entries are not translation from one article rather they are written by different users. Filatova in [32] make use of the information overlap pattern between entries of same article in different languages to device a summarization algorithm. She shows that this information overlap fits in the pyramid summary framework. The pyramid is a representation of the gold-standards summary of a set of documents, representing the opinions of multiple human summary writers who have written summaries for the set of documents. It quantitatively represents the agreement among them. The content of the documents are divided into summary content unit and these units are given weight according to their agreement amongst summarizer and then they are organized into a pyramid of levels where units with similar weight appear in the same level [33].

The author used the list of people created for the Task 5 of DUC 2004 and downloaded Wikipedia entries for each person in all languages and used Google translate to translate non-English entries into English. On average a person from the dataset had description in 25.35 languages. She excluded person's who only have English Wikipedia entries. She then divided the entries using the LingPipe[2] sentence chunker and identified matching sentences from other languages to English using the LingPipe string matching tool. LingPipe is a tool kit for processing text using computational linguistics. This tool is based on TF/IDF distance. The IDF value was computed

[2] LingPipe Http://alias-i.com/lingpipe

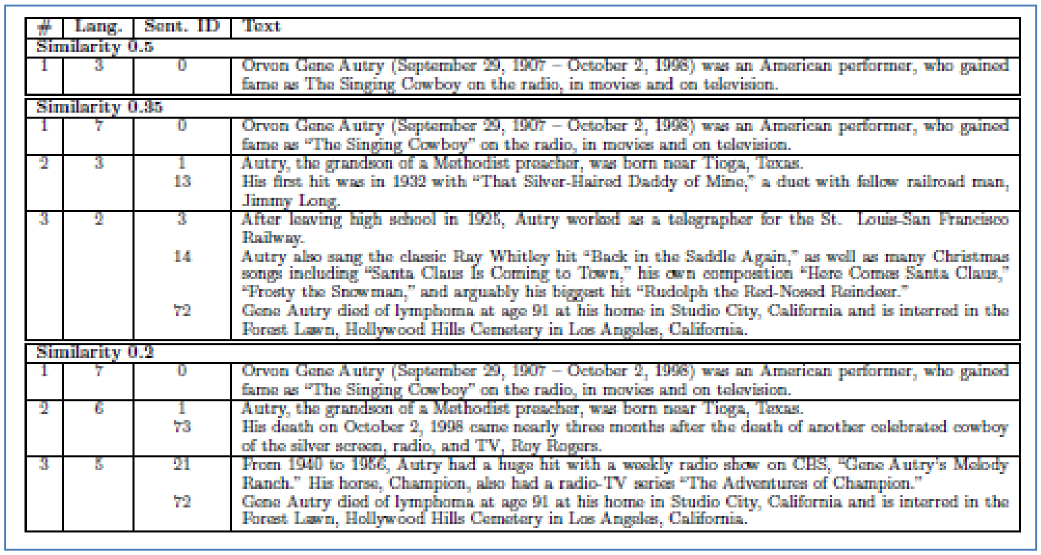
based on the two entity description; the English and non-English one. They used three similarity thresholds: 0.5,0.35,0.2 [32]. They measured the similarity in one person's description in English and in other languages. Their hypothesis was that repeated information in different languages corresponds to the pyramid summarization model. They first add sentence that correspond to the top level of the pyramid; having their counterparts in the most number of descriptions of that person in languages other than English. If the length of the summary is not yet satisfied, they add sentences from the other level; with sentences having the next most number of languages and so on. In their experiments they used the top three levels of the pyramids. Below is the outline of the algorithm used.



	Algorithm
1	Submit the person's name to Wikipedia
2	Get Wikipedia entry descriptions for this person in all possible languages
3	Remove non-plain text information from the descriptions
4	For all the languages handled by the Google MT, translate entry descriptions into English
5	Break English texts into sentences
6	Use a similarity measure to identify what English sentences have counterparts in entry descriptions in other languages
7	Rank all the sentence from the English document according to the number of languages that have similar sentences
8	If several sentences are placed on the same level, list these sentence in the order they appear in the Wikipedia entry description in English
9	Use the top three levels from the above ranking

FIGURE 9 USE OF MULTILINGUAL WIKIPEDIA ALGORITHM OUTLINE [34]

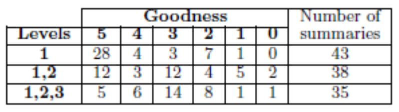
Figure 10 shows a table representing three-level summaries for the English Wikipedia entry of Gene Autry that had entries in 11 languages. Using DUC 2004 set they created one-level summaries for 5 people, two-level summaries for 3 people and three-level summaries for 35 people. For the three-level summary, an average of 3.74 sentences was generated and maximum of 9 sentences summary. The difference between the average and maximum length is due to the length variation of the English Wikipedia entry and the difference in number of other languages entries and their lengths from one person to another [32]. For each output five human annotators were recruited for evaluation purpose. They were asked question regarding summary and were asked to rate (out of five) the goodness of the summary for a particular level given the length constraint. Figure 11 summarizes evaluation results.



#	Lang.	Sent. ID	Text
Similarity 0.5			
1	3	0	Orvon Gene Autry (September 29, 1907 – October 2, 1998) was an American performer, who gained fame as The Singing Cowboy on the radio, in movies and on television.
Similarity 0.35			
1	7	0	Orvon Gene Autry (September 29, 1907 – October 2, 1998) was an American performer, who gained fame as "The Singing Cowboy" on the radio, in movies and on television.
2	3	1 13	Autry, the grandson of a Methodist preacher, was born near Tioga, Texas. His first hit was in 1932 with "That Silver-Haired Daddy of Mine," a duet with fellow railroad man, Jimmy Long.
3	2	3 14 72	After leaving high school in 1925, Autry worked as a telegrapher for the St. Louis-San Francisco Railway. Autry also sang the classic Ray Whitley hit "Back in the Saddle Again," as well as many Christmas songs including "Santa Claus Is Coming to Town," his own composition "Here Comes Santa Claus," "Frosty the Snowman," and arguably his biggest hit "Rudolph the Red-Nosed Reindeer." Gene Autry died of lymphoma at age 91 at his home in Studio City, California and is interred in the Forest Lawn, Hollywood Hills Cemetery in Los Angeles, California.
Similarity 0.2			
1	7	0	Orvon Gene Autry (September 29, 1907 – October 2, 1998) was an American performer, who gained fame as "The Singing Cowboy" on the radio, in movies and on television.
2	6	1 73	Autry, the grandson of a Methodist preacher, was born near Tioga, Texas. His death on October 2, 1998 came nearly three months after the death of another celebrated cowboy of the silver screen, radio, and TV, Roy Rogers.
3	5	21 72	From 1940 to 1956, Autry had a huge hit with a weekly radio show on CBS, "Gene Autry's Melody Ranch." His horse, Champion, also had a radio-TV series "The Adventures of Champion." Gene Autry died of lymphoma at age 91 at his home in Studio City, California and is interred in the Forest Lawn, Hollywood Hills Cemetery in Los Angeles, California.

FIGURE 10 THREE LEVEL SUMMARY OF GENE AUTRY USING DIFFERENT SIMILARITY THRESHOLDS [35]

In 80% of cases of the Level 1 sentences the annotators where happy with the summaries and 70% were happy with the rest of the levels [32].



	Goodness						Number of
Levels	**5**	**4**	**3**	**2**	**1**	**0**	summaries
1	28	4	3	7	1	0	43
1,2	12	3	12	4	5	2	38
1,2,3	5	6	14	8	1	1	35

Table 4: Evaluation results (using Mechanical Turk).

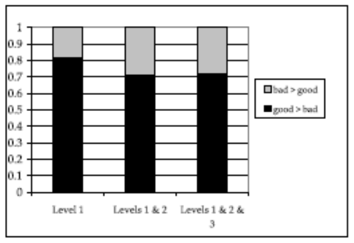
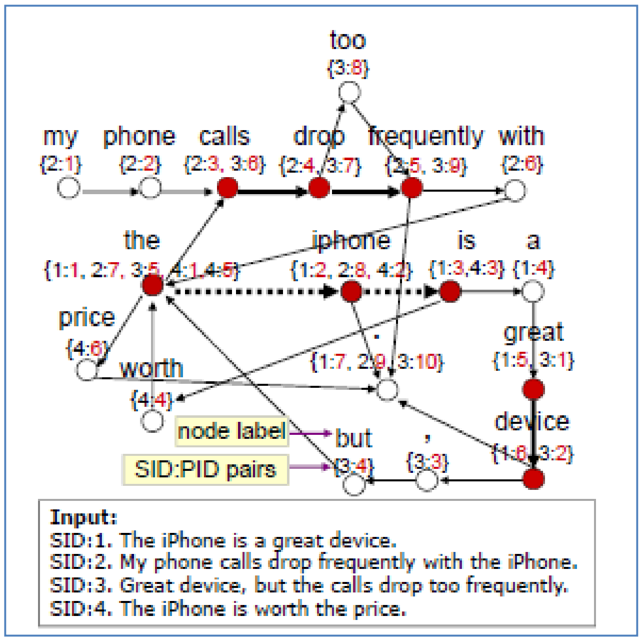


FIGURE 11 MULTILINGUAL WIKIPEDIA SUMMARIZATION EVALUATION RESULTS [36]

2.4.2 *Opinion Summarization Framework: Summarization of Highly Redundant Opinion*

In [2], the authors provide a graph-based summarization framework called Opinosis that is used to generate abstractive summaries of highly redundant opinions. It requires no domain knowledge and makes use of shallow NLP to produce informative abstractive summaries. The main idea behind their work is to first construct a textual graph representing the text to be summarized and using three unique properties of the graph they investigate and score various sub-graphs that are used to generate abstractive summaries.

Their graph structure called Opinosis-Graph is used to represent natural language text and help in finding appropriate paths in the graph to generate the summary. They are different from other extractive graph based approach summaries where those graphs are undirected graphs with sentences as nodes and similarities are edges. However, Opinosis-Graph is a directed graph with word unit as nodes and the edges represent the structure of the sentences and they also attach the positional information of the node. To create Opinosis-Graph they start with the set of sentences to be summarized. Each sentence contains part-of-speech (POS) annotations. Then each sentence is split into word unit that consist of word and its corresponding POS annotation. Each unique word wj will form a unique node vj in the graph having wj as its label. Each node will keep track of all sentences it is part of using a sentence identifier (SID) in addition to position of occurrence in the sentence (PID). Therefore for each node a positional reference information will be created that consist of {SID:PID} pairs that represents the node's membership in a sentence. The structure of the sentence is captured through the use of directed edges to show the sentence [2]. Following is an example of Opinosis-Graph based on four sentences where the thick edges represent salient information.



Input:
SID:1. The iPhone is a great device.
SID:2. My phone calls drop frequently with the iPhone.
SID:3. Great device, but the calls drop too frequently.
SID:4. The iPhone is worth the price.

FIGURE 12 SAMPLE OPINOSIS-GRAPH [37]

From the main properties of Opinosis-Graph that allow it to be used to generate abstractive summaries are redundancy capture, gapped subsequence capture and collapsible structure. The graph capture redundant information through its sub-graph where in the example above, the sentence ‘a great device’ appears to have been mentioned in both sentences 1 and 3. Sentences structure introduces lexical links that makes it easy to discover new sentences or reinforce existing ones. For illustration, in the example above ‘drop frequently’ are mentioned in different ways, this introduce lexical link between ‘drop’ and ‘frequently’ thus ‘too’ can be ignored in sentence 3. This is similar to capturing a repetitive gapped subsequence that shows similar sequences of words with minor variation [2]. In the graph, nodes that connect to various nodes could act like a hub that is possibly collapsible. In the example above, the sub-graph ‘the iPhone is’ is fairly heavy and the node ‘is’ acts like a hub and is a good candidate for compression to generate summary like ‘The iPhone is a great device and is worth the price’ in the example above [2].

The summarization approach they follow aims at repeatedly searching the Opinosis graph for appropriate sub-graphs that encode a valid sentence and have a high redundancy scores thus forming an abstractive summary. The summary is a form of shallow abstractive form since it can only contain words that occur in the text however it has elements of fusion and compression so the sentences generated are not generally the same as the original ones. They define a valid path to be a path that intuitively correspond to meaningful sentences which is defined as follows A path W={vq…vs} is valid if it is connected by a set of directed edges such that vq is a valid start node and vs is a valid end node and W satisfied a set of well-formedness POS constraints [2]. A valid start node is defined as a node that is a natural starting point of a sentence. While a valid end node is a node that completes a sentence where it could be a punctuation such as period or comma or any coordinating conjunction like 'but' or 'yet' [2]. They used the following POS constraints to ensure the valid path is of a well-formed sentence, these rules apply to comparative sentences making it application specific, the filter path so that they match one of those rules so for example for the first rule a sentence must have a noun followed by a verb then followed by adjective with allowing words in between and at the end[2]

1. .* (/nn) +.* (/vb) + .*(/jj) + .*
2. .* (/jj) + .* (/to) + .*(/vb) .*
3. .* (/rb) *. * (/jj) + .* (/nn) + .*
4. .* (/rb) + .* (/in) + .* (/nn) + .*

They give a path a redundancy score of the number of overlapping sentences covered by that path. They also score redundancy weighted by the path length since intuitively longer path will be of more value. In addition, in some cases paths could be collapsible thus a path is collapsed and then scored. Their summarization algorithm is as follows. They first rank all paths in descending order of their scores. Then they eliminate duplicates using a similarity measure namely Jaccard. They then take the top few remaining paths as the generated summary with the number of paths controlled by a parameter that represents the summary size [2]. The algorithm starts by construction of the graph then a depth first traversal of this graph is done to locate valid paths by determining if the node is a valid starting node and if so the algorithm invokes the traverse algorithm that finds the valid path and accumulates its score. Once all paths are explored duplicate paths are removed and the remaining ones are stored in ascending order and the best top candidates are picked as the final summary [2].

For evaluation, they collected reviews from Tripadvisor, Amazon and Edmunds about hotels, cars and various products. Then based on these reviews, two human evaluators were asked to construct opinion seeking queries consisting of entity name and topic of interest for example "Amazon Kindle:buttons" [2]. They compiled a list of 51 queries and created one document per query by collecting all review sentences that contain the query words of the entity name. Each document was of approximately 100 sentences of unordered redundant review sentences. They used the ROUGE system for evaluation that is based on n-gram co-occurrence between machine summaries and human summaries. They used ROUGE-1, ROUGE-2 and ROUGE-N scores. They got five different human workers to summarize each review document and manually reviewed the summaries and got around four reference summaries for each document. Due to limitation in abstractive summarization work, they used MEAD extractive summarizer as a baseline for comparison. They also devised a readability test to evaluate the readability of the generated summaries. They mixed sentences from human summaries with system-generated summaries and asked human judges to pick the most N sentences that are least readable and if they don't often pick the system generated one then they consider the generated summary to be good. The following tables show comparison between Opinosis and human and baseline (MEAD). It shows that Opinosis has closer performance to human summaries than baseline [2]. In addition, using the readability tests they found that 60% of the generated sentences are indistinguishable from the human sentences [2].

TABLE 5 OPINOSIS RECALL ROUGE SCORES [2]

	Recall			
Summary	**ROUGE-1**	**ROUGE-2**	**ROUGE-SU4**	**Avg # words**
Human	0.3184	**0.1106**	0.1293	17
Opinosis	0.2831	0.0853	0.0851	15
Baseline	**0.4932**	0.1058	**0.2316**	75

TABLE 6 OPINOSIS PRECISION ROUGE SCORES [2]

	Precision			
Summary	**ROUGE-1**	**ROUGE-2**	**ROUGE-SU4**	**Avg # words**
Human	0.3434	0.121	0.1596	17
Opinosis	**0.4482**	**0.1416**	**0.2261**	15
Baseline	0.0916	0.0184	0.0102	75

TABLE 7 OPINOSIS F-SCORE ROUGE SCORES [2]

	F-score			
Summary	**ROUGE-1**	**ROUGE-2**	**ROUGE-SU4**	**Avg # words**
Human	0.3088	**0.1069**	**0.1142**	17
Opinosis	**0.3271**	0.0998	0.1027	15
Baseline	0.1515	0.0308	0.0189	75

2.4.3 MULTI-SENTENCE COMPRESSION

In [1], the author aims to compress a cluster of multiple sentences by building a word graph from all words of the sentences and compress that graph by finding the shortest paths form start to end. The advantage of this approach is that it is syntax-lean and requires little more than a tokenizer and a tagger. However, it faces the challenge of selecting important content and providing the compressed sentence in a readable presentation. In the graph she construct, nodes to represent words and edges to represent adjacency between words. To construct the graph, she first add the start and end node, these nodes represent the start and end of the sentences. The start node has an edge to each of the nodes of the starting word of sentences and the end node of each sentence has an edge to the end node. Iterating through sentences, each sentence is divided into words and each word is mapped to the graph taking its part-of-speech into consideration as follows:

- If no node with the word is present in graph then node is added
- If a node has one node with same lowercase word and part-of-speech tag and no word from the same sentence has been mapped to that node then this word is mapped to it, otherwise a new node with same word is added
- If there exists several candidates for the word then the previous and next words are checked for overlap with the sentences or the node with greater frequency mapping is chosen
- Stopwords are mapped if there is an overlap between context meaning that there is an overlap between non-stop neighboring words of the stopword and those to the node otherwise a new node is added for this stopword

Edges are initially added with weight equal to 1 and then incremented by one whenever two words appear adjacent in sentences. Nodes store the id of sentences it came from and their offset position in those sentences. This construction method guarantees that every input sentence corresponds to a loopless path in the graph. Word referring to same entities or actions are likely to end up in one node and stopwords are only joined in one node if there is an overlap in context [1]. Following is an illustration of a sample graph constructed by this method.

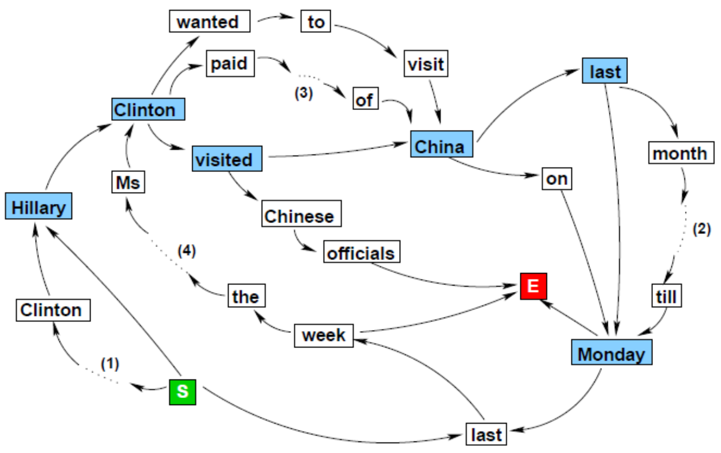


FIGURE 13 WORD GRAPH FOR MULTIPLE SENTENCE COMPRESSION [38]

After graph is constructed, she finds the shortest paths between start and end node for compression through two methods. First method inverts the edge weight and search for shortest path (highest in term of edge weight) from start to end. She sets a predefined minimum length for the path to be 8. Paths are then filtered to remove ones that does not contain a verb. She generates k sentences and filters as above and selects the sentence with minimum total weight. She uses the k-shortest paths algorithm. In the second method, she uses more sophisticated weighting function to generate a grammatical compression that favors strong links (links between words which appear significantly often in that order). As well as, generating an informative compression, it promotes paths passing through salient nodes. She redefines the edge weight to be as follows [1]:

$$w(e_{i,j}) = \frac{freq(i) + freq(j)}{freq(e_{i,j})}$$

This is to give more weight to edges between frequent words. This also promotes connection between two words if there is a multiple paths between them. However, she notes that longer paths between words are weak signals of word association. Therefore, the weight of an edge between the nodes i and j is reduced for every possible path between them but reduced proportionally to its length, so the weight is redefined as follows [1]:

$$w'(e_{i,j}) = \frac{freq(i) + freq(j)}{\sum_{s \in S} diff(s,i,j)^{-1}}$$

diff(s,i,j) refers to the distance between the offset positions of words i and j in sentences s and is defined as follows [1]:

$$diff(s,i,j) = \begin{cases} pos(s,i) - pos(s,j) \;\; if\; pos(s,i) > pos(s,j) \\ 0 \; otherwise \end{cases}$$

To generate a summary concerning the most salient events and entities, she forces the path to go through the most frequent nodes by decreasing the edge weight with respect to the frequency of words its connects, therefore the edge weight is redefined as follows [1]:

$$w"(e_{i,j}) = \frac{w'(e_{i,j})}{freq(i) \times freq(j)}$$

She implement the K-shortest path algorithm to find the fifty shortest paths from start to end using w’’ then she filters the paths that are shorter than eight words and don’t pass through a verb node. Finally, she re-ranks the remaining paths by normalizing the total path weight over its length to obtain the path which has the lightest average edge weight.

She used the Viterbi algorithm to find sequence of word of predefined length n with max bigram probability to act as the baseline for comparison to their approach. The dataset used consisted of news articles presented in clusters on Google News. She got articles in both English and Spanish. She had 10-30 articles per cluster and 24 articles on average. She used the article initial sentence to be their reference summary as it is being accepted as standard competitive baseline in summarization. She stripped off bylines and dates from the beginning of each sentence. She had total of 150 English clusters using 70 for development and 80 for testing.

She used human raters for evaluation where each one is given a set of questions to answer. The questions used were about the quality of the sentence cluster where raters were asked whether the cluster containing a single prevailing event or is it too noisy. Also regarding summary sentence, raters were asked to rate it as perfect if it is a complete grammatical sentence giving it 2 points, almost perfect if required a minor editing giving it 1 point or ungrammatical if none of the above and sentence is given 0 points. Raters were asked to ignore lack or excess of capitalization or punctuation. Regarding informatively, raters could give a cluster a rate of N/A if it is too noisy or perfect if it conveys gist of main event giving it 2 points or rate it as related if it is related to main theme but missed something important and give it 1 point. If not related then it is rated as unrelated and given 0 points. The results of the evaluation are summarized in table below for English and Spanish [1].

TABLE 8 EVALUATION RESULTS FOR ENGLISH/ SPANISH RATING [1]

System	Gram	Info
Baseline	0.70/0.61	0.62/0.53
Shortest path	1.30/1.27	1.16/0.79
Shortest path++	1.44/1.25	1.30/1.25

She did majority voting and resolved ties by assigning lower score and dismissed cases that got ties between min and max. She considered clusters that are classified as containing a single prevailing event by at least ten voters. Following is table that summarizes results distributed over possible rating and average length for English and Spanish [1].

TABLE 9 DISTRIBUTION OVER POSSIBLE RATINGS AND AVERAGE LENGTH FOR ENGLISH AND SPANISH [1]

System	Gram-2	Gram-1	Gram-0	Info-2	Info-1	Info-0	Avg. Len.
Baseline (EN)	21%	15%	65%	18%	10%	73%	8
Shortest path (EN)	52%	16%	32%	36%	33%	31%	10
Shortest path++(EN)	64%	13%	23%	52%	32%	16%	12
Baseline (ES)	12%	15%	74%	9%	19%	72%	8
Shortest path (ES)	58%	21%	21%	23%	26%	51%	10
Shortest path++ (ES)	50%	21%	29%	40%	40%	20%	12

2.4.4 Image Captioning

Summarization could be used in order to provide image captioning. In [39] the authors work on this problem by presenting two different approaches to automatic captioning of geo-tagged images through the summarization of multiple web-documents with information related to the image's location. They used a corpus of 308 images with manually assigned places names and for each image there is up to four short descriptions that are considered as model summaries. They were created manually taken from Virtual Tourist and contain a minimum of 190 words with maximum of 210 words [39]. The top ten web-document returned from Yahoo! Search using the place name as a query is used to generate the image caption automatically through summarization. They built on their work done in [40] that applies shallow preprocessing over the document which include things like sentence detection and POS tagging. Next the summarizer translates the text to WordNet concept to disambiguate the meaning of each term according to its context then the resulting concepts are extended using their hyponyms building a graph representation for each sentence. The vertices represent distinct concepts in the sentence and the edges represent is-a relations. Then all sentence graphs are merged into one single document graph which is extended with a further semantic relation. Each edge is assigned a weight that is directly propositional to the hierarchy of the nodes and the sum of the weights of the edges connected to vertices is calculated to measure the salience of it. The top n vertices are grouped into Hub Vertices Sets that represents set of concepts strongly related in meaning. These vertices are considered the centroids of the clusters. Then a degree-based clustering method is applied over the graph to obtain sub-clusters representing subthemes. Then the clusters with more concepts are computed to be the ones representing main theme and sentences with greater similarity of that clusters are selected in the summary. This technique was designed for single document summarization and to overcome this all content of the documents selected related to image are merged in one document and summarized [39].

Next they build on the work done by [41] that is a statistical-based summarizer. They use three features which are textual entailment that detects redundant information, followed by term frequency and code quantity principle to measure the importance of the sentence in document. The code quantity principle proves that there is a relation between a piece of information and the number of text units it contains. This summarizer again is designed for single document summarization so they used the same way as previous summarizer to tackle this issue. Score is given to each sentence based on these features and top ranking sentences are selected and presented in the order that they appear in text. For a baseline they generate summaries based on the Wikipedia articles describing each image where they select the first 200 words. They used the ROUGE method for evaluation. Their results proved to be satisfactory and not far from those of the Wikipedia summaries [39].

2.5 EVALUATION METHODS

Evaluating the automated summary could be a tricky task. Its difficulty lies in the subjectivity of the summarizing process itself. In human generated summaries, people tend to inject their own knowledge and comments, they are also influenced by their background and knowledge [4]. Two people given the same text to summarize will most probably produce different version of the summary. This in the computational world is very hard to replicate or to automate. We are still very far away from imitating the human knowledge or the complex process of knowledge retention and retrieval done by the human mind. However studies done to further understand how abstractor behave had revealed that professional abstractor tend to use surface-level features like headings and key phrases along with discourse features such as the overall text structure rather than attempting to fully comprehend the text when organizing an abstract [4] [42] . This finding with some adaptation could be modeled in computational sense.

One of the most basic approaches to evaluating automatic summarization is to use human evaluator. Those subjects will be presented with actual test and the summary and provide rating for the output summary. Again the purpose of the summary could also affect the way a human would perceive the goodness or quality of the summary. However, if structured in a clear and concise way and through providing evaluator with clear guidelines for evaluating the output summary, one could provide a reliable way for evaluation. However this is a time and effort consuming way and usually would not scale to a large amount of summaries. To address this problem, Chin-Yew Lin devised a method called ROUGE which stands for Recall-Oriented Understudy for Gisting Evaluation that became the current standard for evaluating the automatic summarization systems. This system aims to measures the quality of a summary by comparing it to other "ideal" human summaries [43]. It counts the number of overlapping units such as n-gram between the computer generated summary and its ideal human counterparts. They introduce four different ROUGE measures: ROUGE-N, ROUGE-L, ROUGE-W and ROUGE-S.

ROUGE-N (where n stand for the length of the n-gram) is an n-gram recall between a candidate summary and set of reference summaries computed as follows [43]

$$\mathbf{ROUGE-N} = \frac{\sum_{s\in\{Reference\ Summaries\}}\sum_{gram_n\in S} \boldsymbol{Count_{match}}\ (\boldsymbol{gram_n})}{\sum_{S\in\{Reference\ Summaries\}}\sum_{gram_n}(\boldsymbol{Count}(\boldsymbol{gram_n})}$$

This is a recall-oriented measure as the denominator of the equation is the total sum of the number of n-grams occurring at the reference summary side. The number of n-grams in the denominator increases as we add more references. The numerator sums over all reference summaries which give more weight to matching n-grams occurring in multiple references. When multiple references are used, pairwise summary-level ROUGE-N between a candidate summary and every references is computed and then take the maximum pairwise score as the final ROUGE-N score.

ROUGE-L applies the longest common subsequent (LCS) measure to summarization. To do so a summary sentence is viewed as a sequence of words where the intuition is that the longer the LCS of two summary sentences is the more similar are the two summaries. They propose using LCS-based F-measure to estimate the similarity between two summaries X of length m and Y of length n as follows [43]:

$$R_{lcs} = \frac{LCS(X,Y)}{m}$$

$$P_{lcs} = \frac{LCS\ (X,Y)}{n}$$

$$F_{lcs} = \frac{(1+\beta^2)R_{lcs}P_{lcs}}{R_{lcs}+\beta^2 P_{lcs}}$$

Advantage of using the LCS is it doesn't require consecutive matches but in-sequence matches that reflect sentence level word order as n-grams. It also automatically includes longest in-sequence common n-grams so no predefined n-gram length is necessary.

To compute the summary level LCS we take the reference summary of u sentences containing a total m words and a candidate summary of v sentences containing total of n words and compute the following [43]

$$P_{lcs} = \frac{\sum_{i=1}^{u} LCS_{\cup}(r_i, C)}{n}$$

$$R_{lcs} = \frac{\sum_{i=1}^{u} LCS_{\cup}(r_i, C)}{m}$$

$$F_{lcs} = \frac{(1+\beta^2)R_{lcs}P_{lcs}}{R_{lcs} + \beta^2 P_{lcs}}$$

ROUGE-W is a weighted LCS measure, it penalizes subsequence matches that are not consecutive [43] [22].

In ROUGE-S, skip-bigram is used, which is any pair of words in their sentence order, allowing for arbitrary gaps. Its co-occurrence statistics measure the overlap of skip-bigrams between a candidate translation and a set of references translation [43]. Skip-bigram F-measure is computes as follows [43]

$$R_{skip2} = \frac{SKIP2(X,Y)}{C(m,2)}$$

$$P_{skip2} = \frac{SKIP2(X,Y)}{C(n,2)}$$

$$F_{skip2} = \frac{(1+\beta^2)R_{skip2}P_{skip2}}{R_{skip2} + \beta^2 P_{skip2}}$$

An advantage to skip-bigram is that it doesn’t require consecutive matches but still sensitive to word order. As opposed to LCS, skip-bigram counts all in-order matching word pairs while LCS only counts the one longest common subsequence. ROUGE-SU is an extension of ROGUE-S by adding the addition of unigram as counting unit to differentiate between similar sentences and words that don't have a single word co-occurrence. Evaluating their method, they found that ROUGE-2, ROUGE-L, ROUGE-W, ROUGE-S worked well for single document summarization task. While ROUGE-1, ROUGE-L, ROUGE-W, ROUGE-SU4 and ROUGE-SU9 worked will in short summaries like headlines. They also found that ROUGE-1, ROUGE-2, ROUGE-S4, ROUGE-S9, ROUGE-SU4 and ROUGE-SU9 worked reasonably for multi-documents summarization when stop-words are excluded from matching. They also found that the exclusion of stop-words improved correlation and correlation to human judgments were increased when using multiple references [43].

2.6 CONCLUSION REMARKS

Most work done in the literature focuses on extractive methods to summarize documents. The abstractive methods proposed rely on advanced Natural Language Processing that sometimes requires domain knowledge. We hope to investigate a method that links extractive and abstractive method without the need for extensive NLP. Sentence compression work proposed in [31][26] promises a way to provide a "shallow" abstractive way to compress sentences into one or two sentences in a way that does not require extensive natural language processing. This method has not to best of our knowledge been used in a full multiple document summarization system before. However, this method has to have sets of sentences in order to compress them. We aim to further investigate the use of a hybrid approach to multiple document summarization that starts with extracting significant sentences from each document in the set of documents to be summarized, clustering them based on similarity, and using sentence compression to provide a compressed version of each cluster that is ordered into the final summary. In the following chapters, we will discuss the details of our research methodology, system design and system evaluation.

CHAPTER 3. RESEARCH METHODOLOGY

Our approach is a hybrid approach of “extractive” and “shallow” abstractive methods that involves several stages of processing and utilizing shallow NLP, not requiring any prior domain knowledge to produce a final summary of multiple documents of the same topic. We first start by selecting significant information presented in sentences of each document, then same or similar sentences are grouped into one cluster to avoid redundancy in information and a compressed version of such sentences are produced for each cluster. Finally we order the produced sentences into a final output summary of the multiple document set. We conducted several research experiments to help us answer our research questions and come up with our system design. The first experiment aimed to identify the best method to use in selecting significant sentences. The second experiment aimed to identify the best measure to use in clustering sentences. The third experiment aimed to evaluate the compressed sentence from each cluster in term of grammatical structure and comprehensibility.

In the first experiment, we investigated the selection of significant sentences using various criteria for scoring sentences. We used three different features which included: presence of high frequent words of document, presence of high frequent words of set of documents and presence of words from first and last sentence of the document. We evaluated the use of each of these features along with all possible combinations as well as using weighted parameters to score sentences. Based on our results, we selected the best performance feature to be used in our system design which was scoring based on presence of high frequent words and presence of words of the first and last sentences in the document. In the next experiment, we clustered similar sentences based on various measures in order to group sentences that mention same or similar content into one cluster. The first measure used word frequency vector to represent sentences in vector space while the second method used word semantics function to represent sentences. In both approaches, we used the k-means algorithm to cluster sentences based on cosine similarity. We compared the clusters produced by using the word frequency vector and word semantics similarity method and chose the best performing approach, which was the word frequency vector. In the last experiment, sentences in each cluster were compressed into one sentence using an approach adopted from the graph-based approach adopted from [1] and [2]. Sentences produced by the compression approach were finally evaluated for grammatical correctness.

3.1. Selecting significant sentences

Objective: In this experiment, we aim to investigate the best method to score sentences based on significance. Significant sentences in a document are sentences that hold most important information that we want to include in the final summary. A way to score that is by finding high frequency terms and measure the presence of such terms in each sentence. Also, highly frequent words of the whole set of documents would indicate topic and important information. Thus, calculating the presence of high frequency words of the whole set of documents would be another way to score sentences. Lastly, an article's initial sentence is considered to provide good summary of the article and some work in the literature considers it to be a standard competitive baseline [1]. Hence, we consider the words in the first sentences as an indication of important information. We also consider words in the final sentence in the keywords set since the last sentence is usually a concluding sentence. Sentences are then scored based on the presence of these words.

Method: We used the three different above mentioned features and combination of them to score sentences for significance to be selected in the sentences to be used in the summary**.** In the first method of high frequent words, the document terms are processed where stop-words are removed and term frequency is measured. The top N frequent words are considered as keywords. Each sentence is then scored for presence of these keywords and the final keywords score is the count of keywords present in the sentence. In the feature for high frequent words of the documents set, we consider the terms frequency in the whole documents set similar to the method above and the top N frequent words are considered as the high frequent words. In a similar fashion, each sentence is scored based on presence of high frequent words. In the final feature, we process the first and last sentences term, remove stop-words and use those terms to give score based on the presence of those words.

We had a sample dataset of 18 documents to select significant sentences from and for each document we had a human reference summary. Through observation and the work presented in the literature, it is shown that comparing to a whole summary is better than comparing to selected sentences. We ran different combinations of features and calculated the ROUGE scores of each set of generated documents and used these scores to select the best performing combination. We also tested having weights for each three features as shown in the score equation below

$$Score(S) = \alpha\, Freq\_Words(S) + \mu\, Set\_Words(S) + \beta\, FirstLast\_Words(S)$$

where $Freq\_Words(S)$ = number of document frequent words present in sentence S, $Set\_Words(S)$ = number of set frequent words present in sentence S and $FirstLast\_Words(S)$= number of words of the first and last sentences of the document present in sentence S.

Following are two experimentations with values of weights that we tried out heuristically:

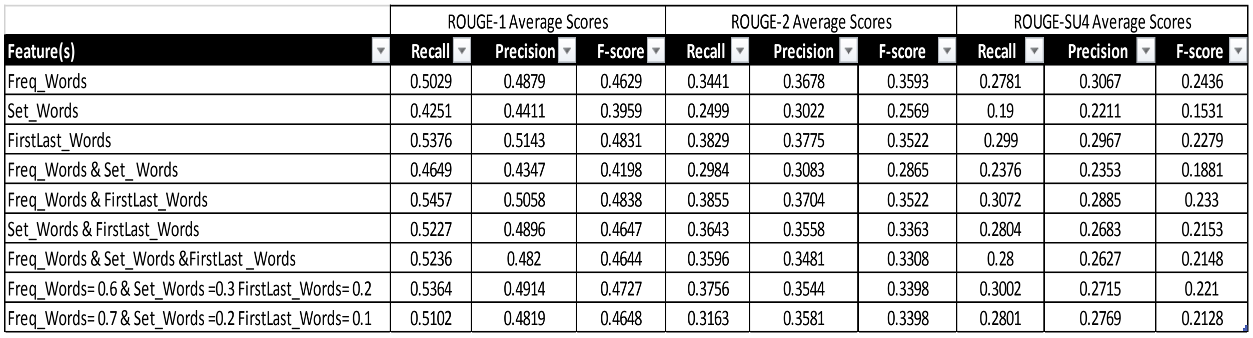
$$\alpha = 0.7, \mu = 0.2, \beta = 0.1$$

$$\alpha = 0.6, \mu = 0.3, \beta = 0.2$$

Following are the different setting of features we tried in our experiment

1. High frequent words of the document feature (Freq_Words)
2. High frequent words of the set of documents feature (Set_Words)
3. First and last sentences words of the document feature (FirstLast_Words)
4. Freq_Words and Set_Words features
5. Freq_Words and FirstLast_Words features
6. Set_Words and FirstLast_Words features
7. All Three Features
8. Weighted three features using $\alpha = 0.7, \mu = 0.2, \beta = 0.1$
9. Weighted three features using $\alpha = 0.6, \mu = 0.3, \beta = 0.2$

Results: We had two summarizers provide us with reference summaries and we compared our generated sentences of summary to those references using ROUGE scores, we calculated ROUGE-1,ROUGE-2 and ROUGE-SU. The following figure shows a table that summarizes the scores of the above mentioned nine configurations.



	ROUGE-1 Average Scores			ROUGE-2 Average Scores			ROUGE-SU4 Average Scores		
Feature(s)	**Recall**	**Precision**	**F-score**	**Recall**	**Precision**	**F-score**	**Recall**	**Precision**	**F-score**
Freq_Words	0.5029	0.4879	0.4629	0.3441	0.3678	0.3593	0.2781	0.3067	0.2436
Set_Words	0.4251	0.4411	0.3959	0.2499	0.3022	0.2569	0.19	0.2211	0.1531
FirstLast_Words	0.5376	0.5143	0.4831	0.3829	0.3775	0.3522	0.299	0.2967	0.2279
Freq_Words & Set_ Words	0.4649	0.4347	0.4198	0.2984	0.3083	0.2865	0.2376	0.2353	0.1881
Freq_Words & FirstLast_Words	0.5457	0.5058	0.4838	0.3855	0.3704	0.3522	0.3072	0.2885	0.233
Set_Words & FirstLast_Words	0.5227	0.4896	0.4647	0.3643	0.3558	0.3363	0.2804	0.2683	0.2153
Freq_Words & Set_Words &FirstLast _Words	0.5236	0.482	0.4644	0.3596	0.3481	0.3308	0.28	0.2627	0.2148
Freq_Words= 0.6 & Set_Words =0.3 FirstLast_Words= 0.2	0.5364	0.4914	0.4727	0.3756	0.3544	0.3398	0.3002	0.2715	0.221
Freq_Words= 0.7 & Set_Words =0.2 FirstLast_Words= 0.1	0.5102	0.4819	0.4648	0.3163	0.3581	0.3398	0.2801	0.2769	0.2128

FIGURE 14 EVALUATING DIFFERENT FEATURES TO SELECT SIGNIFICANT SENTENCES USING ROUGE SCORES

Furthermore we compared the different F-scores of each configuration by calculating the average F-score, minimum and maximum F-scores of each of the three ROUGE scores and we show the results of our analysis in the next table

Feature(s)	Average F-score	Min F-Score	Max F-Scor
Document Frequent Words(Freq_Words)	0.355	**0.2436**	0.4629
Document Set Frequent Words (Set_Words)	0.269	0.1531	0.3959
First And Last Sentence Words (FirstLast_Words)	0.354	0.2279	0.4831
Freq_Words & Set_ Words	0.298	0.1881	0.4198
Freq_Words & FirstLast_Words	**0.356**	0.233	**0.4838**
Set_Words & FirstLast_Words	0.339	0.2153	0.4647
Freq_Words & Set_Words &FirstLast _Words	0.337	0.2148	0.4644
Weighted Features (Freq_Words= 0.6 & Set_Words =0.3 FirstLast_Words= 0.2)	0.345	0.221	0.4727
Weighted Features (Freq_Words= 0.7 & Set_Words =0.2 FirstLast_Words= 0.1)	0.339	0.2128	0.4648

FIGURE 15 COMPARING DIFFERENT F-SCORE TO SELECT METHOD TO USE IN SELECTING SIGNIFICANT SENTENCES

Following are charts showing the various F-score of the nine configurations starting with the F-scores from ROUGE-1, ROUGE-2 and ROUGE-SU4 measures followed by a chart that shows the average F-score of the three scores for each configuration. Another chart shows the minimum F-score of the three scores for each configuration followed by a further chart that shows the maximum F-score of the three scores for each configuration.

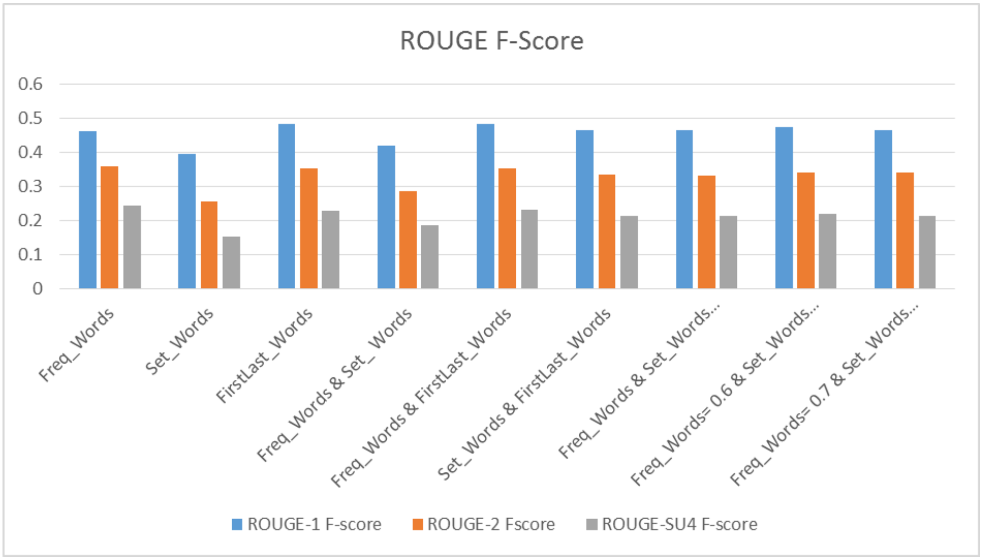


FIGURE 16 F-SORES OBTAINED WHEN RUNNING DIFFERENT CONFIGURATION USING ROUGE-1, ROUGE-2 AND ROUGE-SU4

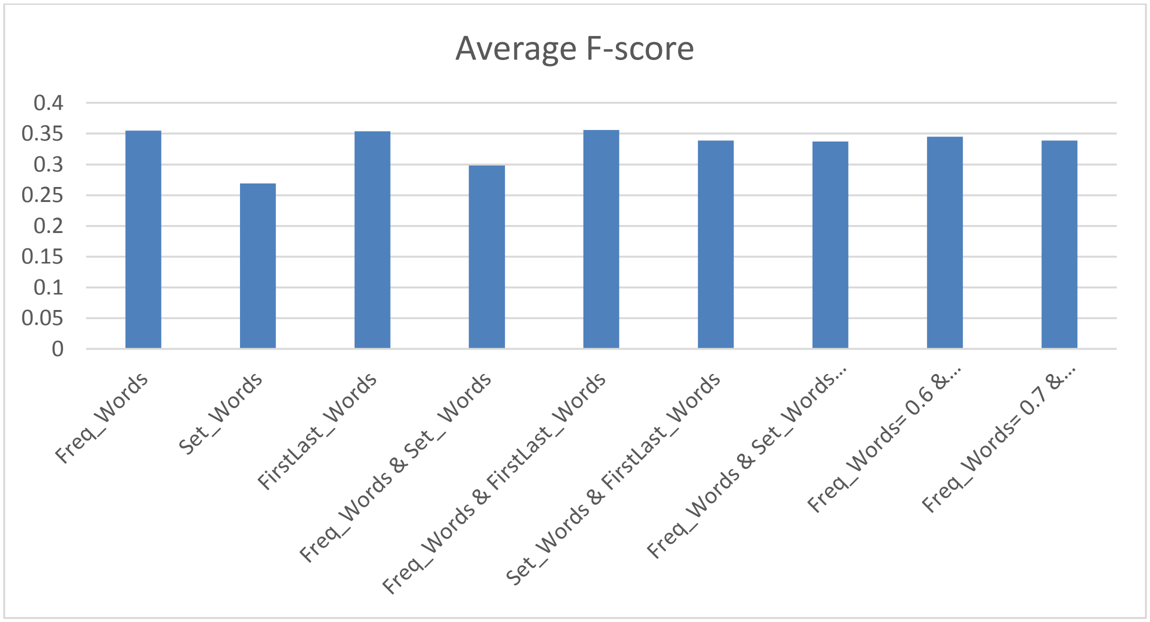


FIGURE 17 AVERAGE F-SCORE OF THE THREE ROUGE MEASURES USED AS EVALUATING MEASURES FOR SELECTING SIGNIFICANT SENTENCES

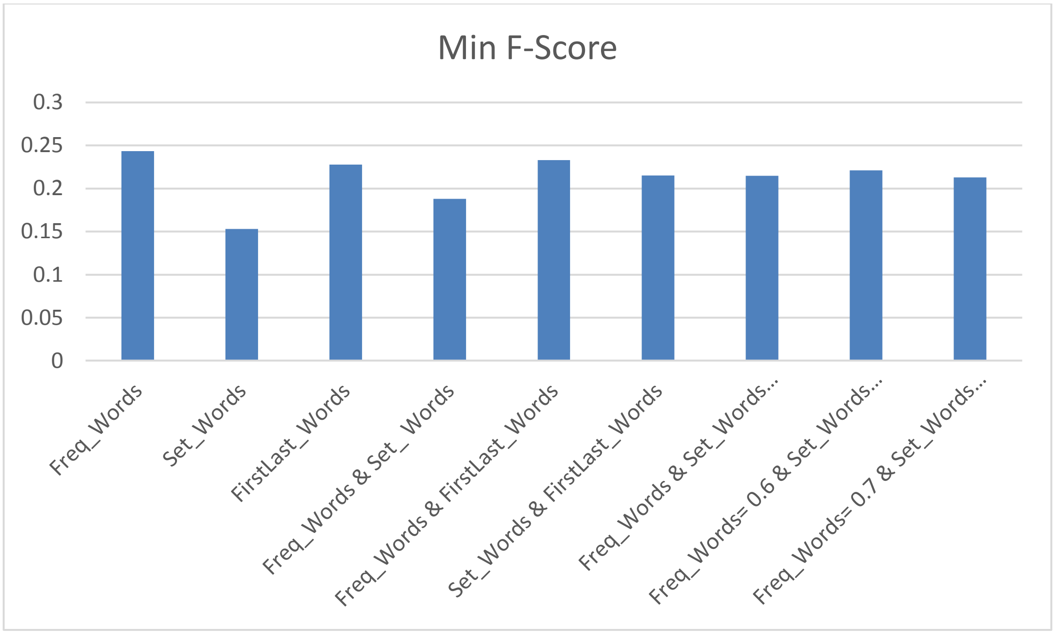


FIGURE 18 MINIMUM F-SCORE OF THE THREE ROUGE MEASURES USED AS EVALUATING MEASURES FOR SELECTING SIGNIFICANT SENTENCES

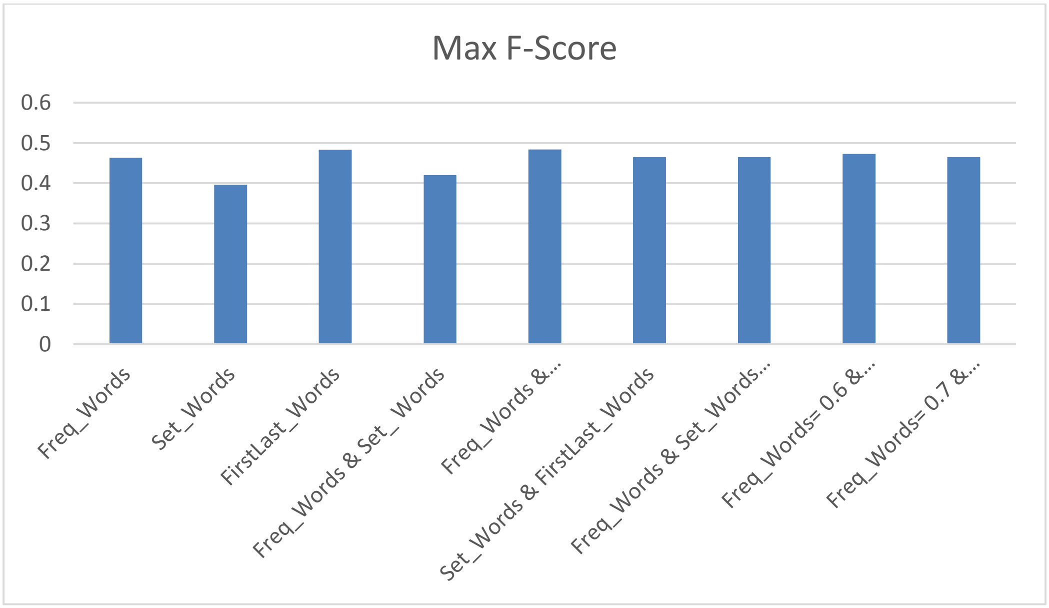


FIGURE 19 MAXIMUM F-SCORE OF THE THREE ROUGE MEASURES USED AS EVALUATING MEASURES FOR SELECTING SIGNIFICANT SENTENCES

Discussion:

As shown in the results, we used the f-score to compare between the different configurations scores. The use of the document high frequent words and words of the first and last sentences provided the best f-score when comparing average and maximum f-score of different ROUGE measures. In addition, the use of the document high frequent word provided the best minimum f-score. So by voting, we selected the use of the high frequent words of the document and the words of the first and last sentences to be considered as our measure for scoring sentences to select significant sentences. We considered the above features because we believed they are the most promising features to score sentences based on significance. Other features proposed in the literature make use of words appearing in sentences that appears in titles or come after main headers however since our dataset does not have headers or titles we did not use such methods. We also investigated the use of cue words. However, through our research we found that it is a genre-dependent feature [44] and in one work investigation of news articles it didn't yield cue words [44]. Thus, we opted against using such feature. The use of weighted parameters proposed in our configuration above was selected heuristically based on our judgment of the importance of each feature. Since we don't have any tagged data that we can utilize to try to optimize those parameters and the equal weighted features provided promising results we did not investigate further the weighted parameter configuration. Since that we utilized human summarizer the number of documents we had was the ones we obtained from them.

3.2. SENTENCES SIMILARITY MEASURE USED IN CLUSTERING

Objective: In this experiment, we aim to select best method to measure sentences similarity to be used in clustering. We aim to cluster similar sentences into one cluster to avoid redundancy in information. We investigate two methods. We first use the word term frequency measure to create clusters of sentences. We use cluto[3] toolkit based on cosine similarity. Cluto toolkit is a software

[3] http://glaros.dtc.umn.edu/gkhome/cluto/cluto/overview

package for providing clustering functionality for high and low dimensional datasets. We use the vcluster program provided with cluto which is used to cluster dataset represented in the vector space. It uses a bisecting k-means algorithms to accomplish the task of clustering. We selected the use of such tool since that it is publicly available and easy to use and also provides a tool to accomplish the task of representing documents in the vector space. We then use the word semantic similarity method mentioned in our approach above to produce clusters of same sentences. We use the scluster program provided with cluto which is used to cluster dataset represented in similarity space. We then evaluate clusters of both measures and select the best measure to use in our approach.

Method: The first approach considers the term matrix, where a vector of term frequency is computed for each sentence. Then we run the cluto clustering toolkit to create the word vector for each sentence and then cluster sentences based on cosine similarity between each vector.

In the second approach, we adopted the algorithm used in [7] on a Chinese dataset. The similarity between two sentences is measured as follows [7]:

$$Sim(S_1, S_2) = \lambda_1 * \mathrm{Sim1}(\mathrm{S}_1, \mathrm{S}_2) + \lambda_2 * \mathrm{Sim2}(\mathrm{S}_1, \mathrm{S}_2)$$

Where λ_1 = 0.3 and λ_2= 0.7. Sim1 measures the word form similarity between two sentences in which the number of same words that appears in both sentences is computed. Stop-words and punctuations are removed before computing this similarity. Thus, Sim1 is computed between two sentences S1 and S2 as follows [7]:

$$Sim1(S_1, S_2) = 2 * (\mathrm{SameWord}(\mathrm{S}_1, \mathrm{S}_2)/(\mathrm{Len}(\mathrm{S}_1) + \mathrm{Len}(\mathrm{S}_2)))$$

SameWord(S_1,S_2) computes the number of the same words in two sentences and Len(S) represents the word number in the sentence S.

Sim2 computes the word semantic similarity between sentences S1 and S2. The semantic is measured based on WordNet [4] which is a lexical database for English language. In WordNet, nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms called synsets. Each sysnset express a distinct concept. They are interlinked by means of conceptual-semantic and lexical relations. We use RiTa.WordNet[5] which is a WordNet library for Java. Word-Sentence Similarity (WSSim) is defined to be the maximum similarity between the word w and words within the sentence S [7].

$$WSSim(w, S) = \max\{Sim(w, W_i) \mid W_i \in S, where\ w\ and\ W_i\ are\ words\}$$

Here, the Sim(w,W_i) is the word similarity between w and W_i. We compute the similarity based on the distance function in the RiTa library which measure the semantic distance between two words so if two words are same their distance is zero and we measure their similarity to be

$$Sim(w_1\ , w_2) = 1 - distance\ (w_1, w_2)$$

Thus, the Sim2 between sentences S1 and S2 is computed as follows [7]:

$$Sim2(S_{1,}S_2) = \frac{\sum_{wi \in S1} \text{WSSIM(wi,S2)} + \sum_{wj \in S2} \text{WSSIM(wj,s1)}}{|S1|+|S2|}$$

After computing the semantic similarity between the sentences, we run cluto toolkit for clustering based on the cosine similarity between similarity vector for each sentence where each feature of the vector constitutes the similarity between the sentence and other sentences. Computing the k

[4] http://wordnet.princeton.edu/
[5] http://www.rednoise.org/rita/wordnet/documentation/

we will adopt the measure proposed in [7] and compute k with a threshold to be no more than the average of length of individual articles in the set of documents as a whole. Following is the formula used in [7]:

$$k = n\frac{|D|}{\sum_{i-1}^{n}|Si|} = n\ \frac{|\bigcup_{i-1}^{n} Si|}{\sum_{i-1}^{n}|Si|}$$

Where |D| is the number of terms in the document D, |Si| is the number of terms in Sentences Si and n is number of sentences in Document D.

Results: Using 32 different clusters produced by each of the two methods, we scored the clusters based on the similarity of sentences to give them 0 for mostly un-similar sentences, 1 for somewhat similar sentences and 2 for perfectly or near-perfectly similar sentences. Following are the percentage of each of those scores for both methods, Figure 35 represents scores for the word vector method and Figure 36 represents scores for the word semantics method.

TABLE 10 EVALUATION SCORES FOR CLUSTERS PRODUCED BY WORD FREQUENCY VECTOR METHOD USING 3 SCALE SCORES

Score	Percentage
0	9.375
1	43.75
2	46.875

TABLE 11 EVALUATION SCORES FOR CLUSTERS PRODUCED BY WORD SEMANTICS METHOD USING 3 SCALE SCORES

Score	Percentage
0	25
1	62.5
2	12.5

Discussion:

Using semantic similarity to cluster sentences initially had promise since it does not just rely on the same word for similarity. However, the semantic similarity is measured between two words taking in consideration its part of speech and a word could have different meaning depending on the context of the sentence in which it appears. Thus semantic similarity measure could be off in that the words could have semantic similarity but the way one word is used in its sentence could change that semantics without the measure being able to measure that change and would consider the sentences to be similar while they are not. Thus through our evaluation and observation we found that the word term frequency vector measure produced clusters of better quality in term of sentences similarity where the cluster of 2 and 1 scores have higher percentage than that of the other measure. We selected an arbitrary number of clusters at random to try to make the evaluating sample as representative as possible of the clusters.

3.3. SENTENCE COMPRESSION OF CLUSTERS

Objective: In this experiment, we ran the graph-based approach to compress each cluster into one sentence. Since that similar sentences are clustered into one cluster, compressing them into one sentence will avoid redundancy in information as much as possible. Human evaluators evaluated the compressed sentence by giving them score of "2" for perfect sentences that are grammatically correct; a score of "1" for sentences that require minor editing; "0" for incorrect sentences that are none of the above. This helped us evaluate the compressed sentence as a form of "shallow" abstract of the cluster of sentences.

Method: Each cluster of sentences is processed to create a graph representing the words and connection between them. The nodes represent words and the edges represent sentences structure.

When two words appear together more than once, the weight of their corresponding edge is increased by how much they appear together. The edges' weights are then inverted and divided by the product of the two words of the nodes frequency. The graph is then searched for k-shortest paths from start to end node and paths are filtered and ranked according to score of their edges weights. The top path is selected to represent the final compressed sentence. Further details on this approach is provided in the system design in chapter four below.

Results: The following table shows a summary of the results given by three judges for 51 random compressed sentences.

TABLE 12 PERCENTAGE OF EACH SCORE GIVEN BY THREE JUDGES TO SCORE COMPRESSED SENTENCES

Score	Judge A (%)	Judge B (%)	Judge C (%)
0	7.69	5.88	11.54
1	19.23	19.61	11.54
2	73.08	74.51	76.92

Discussion:

These results show that the majority of sentences produced by compression are well structured or require minor editing where sentences scoring perfect score of 2 constituted almost 74% of the evaluated set of sentences. The way the graph is constructed aim to map the words with common context to the same node in order to capture redundancy of information and allow for compression of cluster sentences in a way that would capture the common information in those sentences. We picked a random sentence from each cluster (with an extra sentence) to have a representative sample of the generated summaries of the whole corpus.

CHAPTER 4. SYSTEM DESIGN

In this chapter we go into details about our system design. We start by an overall system design where we explain the various stages of processing our system and then we go into details about each stage explain the processing and the algorithm we used.

4.1. OVERALL SYSTEM DESIGN

Our approach consists of several phases of processing, with the aim of providing a final summary of multiple documents of the same topic. We start with a set of documents of the same topic from which we will extract the significant sentences from each text by scoring each sentence based on the presence of high frequent words of the document and words of the first and last sentences of the document. Sentences are then ranked according to the score and top sentences are selected based on compression criteria. We then use the word frequency vector approach to cluster sentences into similar sentences clusters. Then, each sentence cluster is converted into a graph of words where the edges represent the connection between these words. From this, a sentence is produced as a form of compressed sentence of the clustered content. Finally, we order the compressed sentences into the final summary. To the best of our knowledge, this combination of steps for summarization is novel and the graph-based approach has never been applied in a complete system for multiple document summarization; it has only been used in [1] and [2], to compress sentences and provide summarization for highly redundant opinions. We can envision our system process as follows:

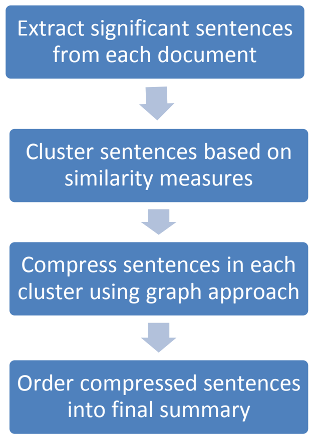

Extract significant sentences from each document
Cluster sentences based on similarity measures
Compress sentences in each cluster using graph approach
Order compressed sentences into final summary

FIGURE 20 SUMMARIZATION SYSTEM PROCESS

The next figure illustrates our system design starting with a set of documents to summarize and ending with the final summary.

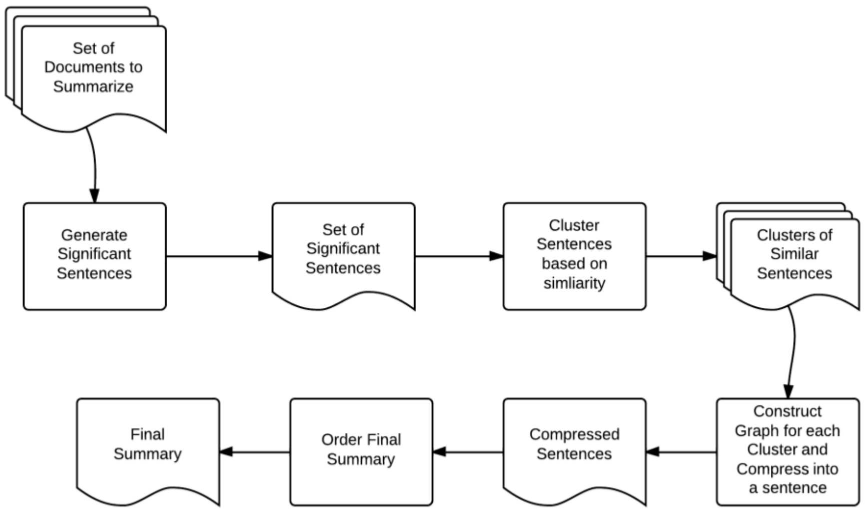

Set of Documents to Summarize
Generate Significant Sentences
Set of Significant Sentences
Cluster Sentences based on simliarity
Clusters of Similar Sentences
Construct Graph for each Cluster and Compress into a sentence
Compressed Sentences
Order Final Summary
Final Summary

FIGURE 21 SUMMARIZATION SYSTEM DESGIN

We built our system using the Java programming language using NetBeans 7.12 IDE. We utilized the JUNG[6] library which the Java Universal Network/Graph library. JUNG is a software library that provides functionality for the modeling, analysis and visualization of data represented in graph. It support directed and undirected graphs. It provides functionality to construct and annotate graphs. It also support visualization of graph that we used in our visualization toolkit we developed to illustrate the constructed graph and the path selected of compressed sentence. Our algorithm takes around 78 seconds to process a cluster of 10 documents with average length of 16 sentences ranging from 2 – 47 sentences long articles

4.2. EXTRACT SIGNIFICANT SENTENCES FROM EACH DOCUMENT IN THE SET

We will start by running the following algorithm to extract the significant sentences:

1. For each sentence in the text, we will extract terms and remove stop-words and punctuations.
2. We calculate the frequency of each term and select the top N words to be the set of high frequent words of the document as well as processing first and last sentences of the document for the set of words to compare to.
3. We will score the sentence based on occurrence of top frequent words as well as occurrence of words appearing in first and last sentence:

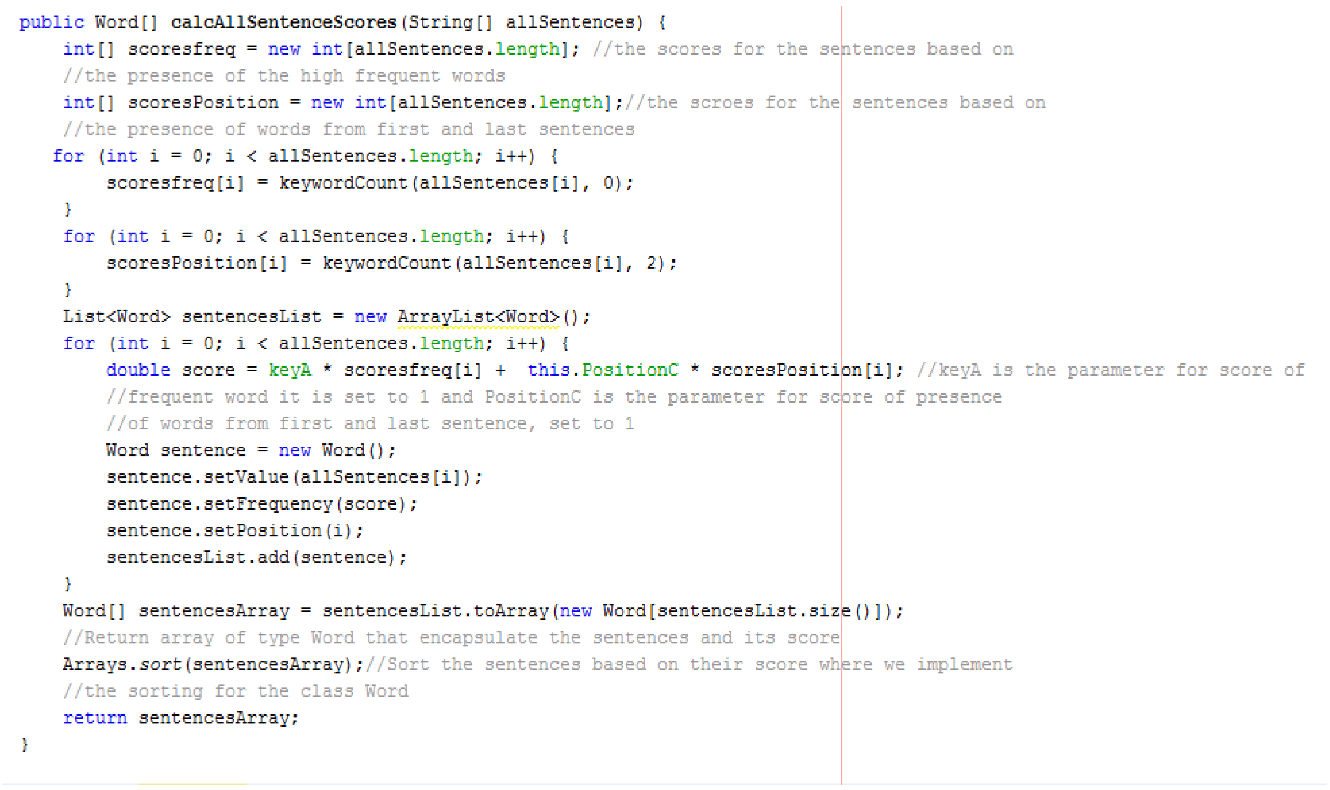
$$Score(S) = Freq\_Words(S) + FirstLast\_Words(S)$$

   where Freq_Words(S) is the sum of words occurring in sentence S from the document high frequent words and FirstLast_Words(S) is the sum of words occurring in the sentence S from the set of words of the first and last sentences of document.

[6] http://jung.sourceforge.net/

4. We then rank each sentence based on its score and retrieve the top N sentences. We measure N in accordance to 30% compress ratio. So, if we have an article of 100 sentences we select top scoring 30 sentences.

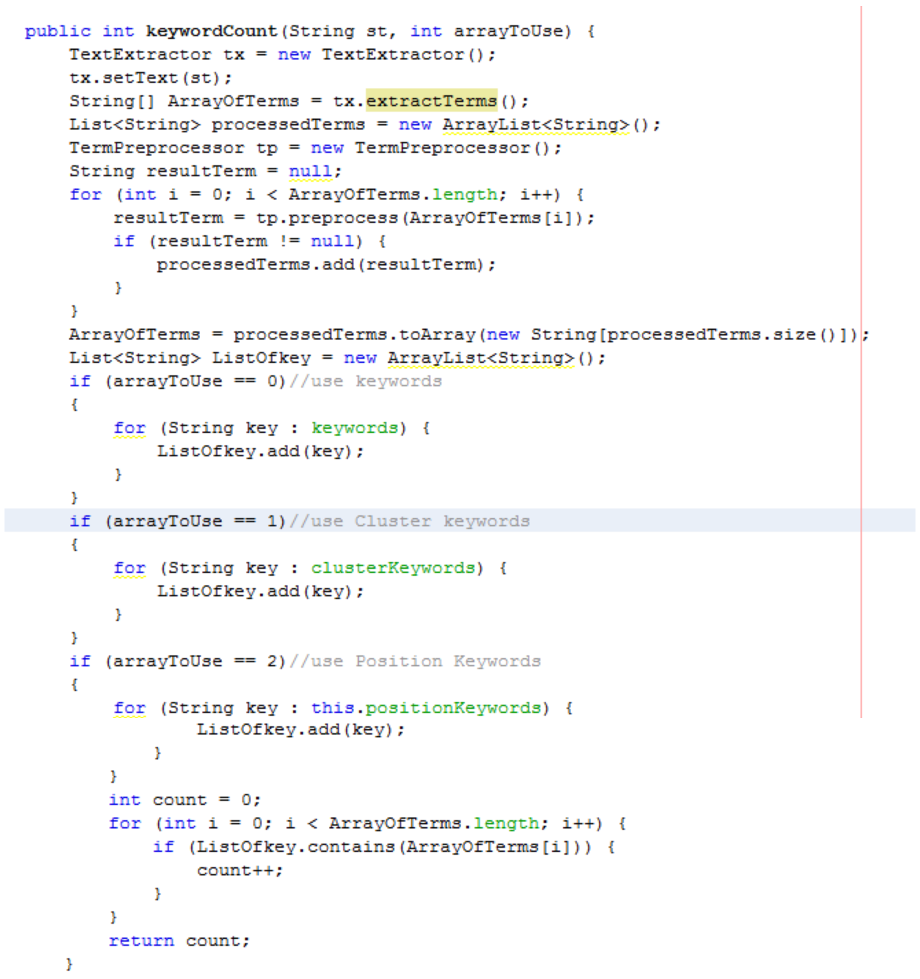
Following is the code for scoring sentences that we used in our system, we set the parameter for the whole document frequent words to zero in order to exclude it from the score.



```java
public Word[] calcAllSentenceScores(String[] allSentences) {
    int[] scoresfreq = new int[allSentences.length]; //the scores for the sentences based on
    //the presence of the high frequent words
    int[] scoresPosition = new int[allSentences.length];//the scroes for the sentences based on
    //the presence of words from first and last sentences
   for (int i = 0; i < allSentences.length; i++) {
        scoresfreq[i] = keywordCount(allSentences[i], 0);
    }
    for (int i = 0; i < allSentences.length; i++) {
        scoresPosition[i] = keywordCount(allSentences[i], 2);
    }
    List<Word> sentencesList = new ArrayList<Word>();
    for (int i = 0; i < allSentences.length; i++) {
        double score = keyA * scoresfreq[i] +  this.PositionC * scoresPosition[i]; //keyA is the parameter for score of
        //frequent word it is set to 1 and PositionC is the parameter for score of presence
        //of words from first and last sentence, set to 1
        Word sentence = new Word();
        sentence.setValue(allSentences[i]);
        sentence.setFrequency(score);
        sentence.setPosition(i);
        sentencesList.add(sentence);
    }
    Word[] sentencesArray = sentencesList.toArray(new Word[sentencesList.size()]);
    //Return array of type Word that encapsulate the sentences and its score
    Arrays.sort(sentencesArray);//Sort the sentences based on their score where we implement
    //the sorting for the class Word
    return sentencesArray;
}
```

FIGURE 22 IMPLEMENTATION THE OF FUNCTION TO SCORE SENTENCES BASED ON PRESENCE OF DOCUMENT HIGH FREQUENT WORDS AND WORDS FROM FIRST AND LAST SENTENCES

The function takes as its input the array of sentences from the document to be scored. It iterate through the sentences to calculate the score for each sentence based on the set of words to score according. It calls the keywordCount function that takes the sentence to be score as well as parameter for which set of words to score against. We use 0 for the high frequent words of the document and 2 for words from the first and last sentences. The first loop in the function will run number of times according to the number of words in set of document high frequent words and the second loop will run according to the number of words in the first and last sentences. It then will loop according to the number of sentences in the document and for each sentences call the keywrodCount function. The implementation of keywordCount function is as follows



```
public int keywordCount(String st, int arrayToUse) {
    TextExtractor tx = new TextExtractor();
    tx.setText(st);
    String[] ArrayOfTerms = tx.extractTerms();
    List<String> processedTerms = new ArrayList<String>();
    TermPreprocessor tp = new TermPreprocessor();
    String resultTerm = null;
    for (int i = 0; i < ArrayOfTerms.length; i++) {
        resultTerm = tp.preprocess(ArrayOfTerms[i]);
        if (resultTerm != null) {
            processedTerms.add(resultTerm);
        }
    }
    ArrayOfTerms = processedTerms.toArray(new String[processedTerms.size()]);
    List<String> ListOfkey = new ArrayList<String>();
    if (arrayToUse == 0)//use keywords
    {
        for (String key : keywords) {
            ListOfkey.add(key);
        }
    }
    if (arrayToUse == 1)//use Cluster keywords
    {
        for (String key : clusterKeywords) {
            ListOfkey.add(key);
        }
    }
    if (arrayToUse == 2)//use Position Keywords
    {
        for (String key : this.positionKeywords) {
             ListOfkey.add(key);
            }
        }
        int count = 0;
        for (int i = 0; i < ArrayOfTerms.length; i++) {
            if (ListOfkey.contains(ArrayOfTerms[i])) {
                count++;
            }
        }
        return count;
    }
```

FIGURE 23 IMPLEMENTATION OF THE FUNCTION TO COUNT THE OCCURENCE OF SPECIFIED SET OF WORDS IN THE SENTENCE

The function takes as a parameter the sentence to be scored and the selection of which set of words to score against. It then count the number of occurrence of such words in the sentence and return this number. All words are processed to remove punctuations and stop-words and make the words lowercase for comparison. The function will loop according to the number of words in the set being considered for scoring and number of terms from the sentence.

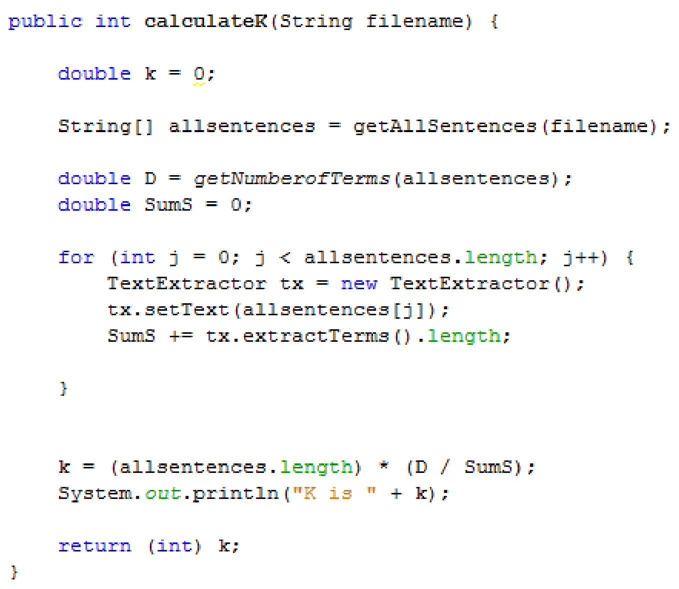
4.3. Cluster sentences based on similarity measures

The extracted significant sentences are clustered based on their similarity. We use term frequency matrix to cluster sentences. Each sentence in the cluster is processed to count the occurrence of each term of the cluster sentences in it and form a word frequency vector. We use cluto clustering toolkit[7] to calculate the word frequency matrix and to cluster the sentences. We calculate K as number of clusters to produce based on the formula used in [7]:

$$k = n\frac{|D|}{\sum_{i-1}^{n}|Si|} = n\,\frac{|\bigcup_{i-1}^{n} Si|}{\sum_{i-1}^{n}|Si|}$$

Where |D| is the number of terms in the document D, |Si| is the number of terms in Sentences Si and n is the number of sentences in Document D. The implementation of the calculation of the k is shown in Figure 24 . The function takes as a parameter the path to the file that contains the set of selected significant sentences from the previous phase of processing. It process the sentences to get the number of terms in it and then calculate the length of each sentence and then calculate K by dividing the sum of terms in the document with the sum of sentences' length and multiply it be the number of sentences in the whole set. The main loop number of runs will be determined by the number of selected significant sentences and for each sentence it will loop according to the length of that sentence.

[7] http://glaros.dtc.umn.edu/gkhome/cluto/cluto/overview



```
public int calculateK(String filename) {

    double k = 0;

    String[] allsentences = getAllSentences(filename);

    double D = getNumberofTerms(allsentences);
    double SumS = 0;

    for (int j = 0; j < allsentences.length; j++) {
        TextExtractor tx = new TextExtractor();
        tx.setText(allsentences[j]);
        SumS += tx.extractTerms().length;

    }


    k = (allsentences.length) * (D / SumS);
    System.out.println("K is " + k);

    return (int) k;
}
```

FIGURE 24 THE IMPLEMENTATION OF THE FUNCTION THAT CALCULATE THE NUMBER OF CLUSTERS TO BE USED IN CLUSTERING

4.4. COMPRESS SENTENCES IN EACH CLUSTER USING GRAPH APPROACH

After sentences are clustered, we compress the sentences in each cluster as follows. We adopt similar approaches to [1] [2]. We selected this method because we believe it is a promising way to provide a "shallow" abstract of the sentences in the cluster and compress them in a fashion that would capture their main information. Words in sentences are tagged by part-of-speech tags using OpenNLP part-of- speech tagger[8]. OpenNLP library is a machine learning based toolkit for natural language processing. It provides most common NLP tasks like tokenizing, parsing and part of speech tagging. Part of speech tagging is the process by which a software process text and assign part of speech tag to it (pos) such as noun, verb, adjective..etc. The Part of Speech Tagger of the OpenNLP tags words with their corresponding part of speech (pos) based on the word itself and

[8] http://opennlp.apache.org/

the context of the word. A word might have multiple pos tags depending on the word and the context. The OpenNLP POS Tagger uses a probability model to predict the correct pos tag out of the tag set. We used an available model for English language. Each cluster of sentences is processed to create a graph representing the words and connection between them. The algorithm goes as follows:

1. Iterating through sentences, we add a start node to represent the start of sentence and an end node to represent the end of sentence.
2. Each node represents a word that appears in a sentence. All words are processed to make it lowercase and remove punctuations. We then check if there exists a node in the graph that has the same part-of-speech tag and no word from this sentence has been mapped to it. If so, we map it to this node if they have the same previous word. If they do not share same previous word we check for same next word in the graph. If the previous node is a stop-word, we check for next word. If next word is also a stop-word then to map to this node, the previous and next nodes' words should be the same to the ones appearing in the sentence of the node being mapped. If no node exists for this word, we add a new node with this word.
3. Each node encapsulates the information about the word, which is: the word text, the sentences it belongs to, its position in this sentence, the frequency of the word in the whole set of documents as well as the order of the sentence in the article.
4. If a word appears twice in a sentence then each word is mapped to a different node to avoid having loops in the graph.
5. Stop words are mapped if there is an overlap in the surrounding nodes in both sentences, so that they have the same previous and next word.

6. Words that has part of speech tag “CD- Cardinal Number” are treated like stop words where they are mapped to node only If they have the same previous and next words.
7. In case of repetitive adjectives we remove one of them to avoid having loops in the path. For example, in a sentence like “it had a tremendous tremendous effect “, we only map one word of “tremendous” since it doesn’t affect the meaning.
8. Edges are added, initially with a weight equal to one, and incremented whenever the same edge between same words is encountered.
9. If a word has two possible nodes to be mapped to, we resolve the mapping based on the surrounding nodes to the candidate nodes. We first look for the node before and after the node to be mapped to and check if they are the same to either the words before and after the word being mapped. If so, then the word is mapped to this node.

Following this method, every sentence in the cluster should be mapped to a loop-less path from a start node to an end node in the graph. Words referring to the same entities or meaning will more likely end up mapped to the same node. Stop-words will be mapped to the same node if there is overlap in context. The weight of the edges will capture the redundancy in information and thus could be used to identify important content that should be included in the final summary.

To compress the sentences, we find the shortest paths from the start node to the end node based on the edge weight. We invert the edge weight and search for shortest paths with a set minimum sentence length. We also incorporated the frequency of the words in the nodes of the edge to the weight by using the formula proposed in [1] where we divide the weight of each edge with the multiplication of the frequencies of each word of its vertices in order to reward edges with most frequent words [1]:

$$w(e_{i,j})' = \frac{1}{w(e_{i,j})}$$

$$w(e_{i,j})'' = \frac{w(e_{i,j})'}{freq(i) \ * \ freq(j)}$$

$$w(e_{i,j}) \ is \ the \ weight \ of \ edge \ between \ node \ i \ and \ j$$
$$freq(i) is \ the \ frequency \ of \ word \ i$$
$$freq(j) is \ the \ frequency \ of \ word \ j$$

The frequency of stop-words are set to 1. The minimum sentence length is set to be 8 and maximum length to be no more than double to the average sentence length in the cluster. We remove sentences with no verbs. We also remove sentences that starts with conjunctions like “but” and “since” since their meaning could depend on a previous sentence that might not appear in text of final summary. We then generate the top N sentences to be the compressed sentences version of the cluster and select the top sentence to act as the compressed version of the sentences in the cluster. Following is an example of compressing four sentences and the graph that we produced using an illustration tool we developed those sentences are:

1. Jose Saramago became the first writer in Portuguese to win the Nobel Prize for literature on Thursday.
2. The Portuguese writer Jose Saramago won the Nobel Prize for literature.
3. Jose Saramago who is 75 and portuguese won the Nobel Prize for literature.
4. While attending the Frankfurt book fair on Thursday, Jose Saramago got the news that he won the Nobel Prize for literature.

The final sentence after compression is: *Jose saramago won the nobel prize for literature*. The following image shows the constructed graph and the path selected for the final sentences in green.

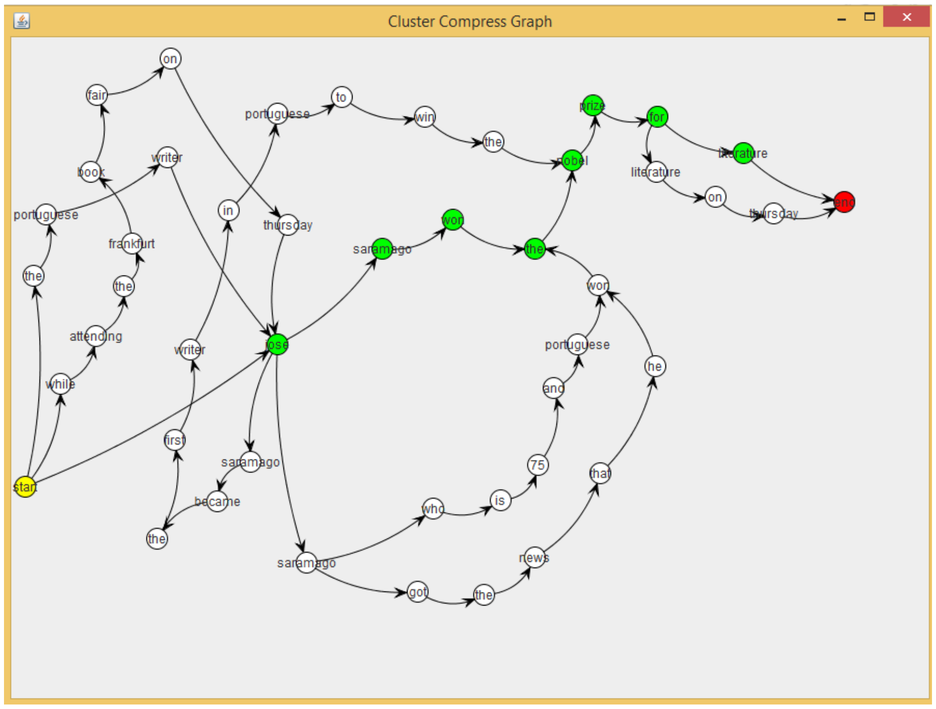
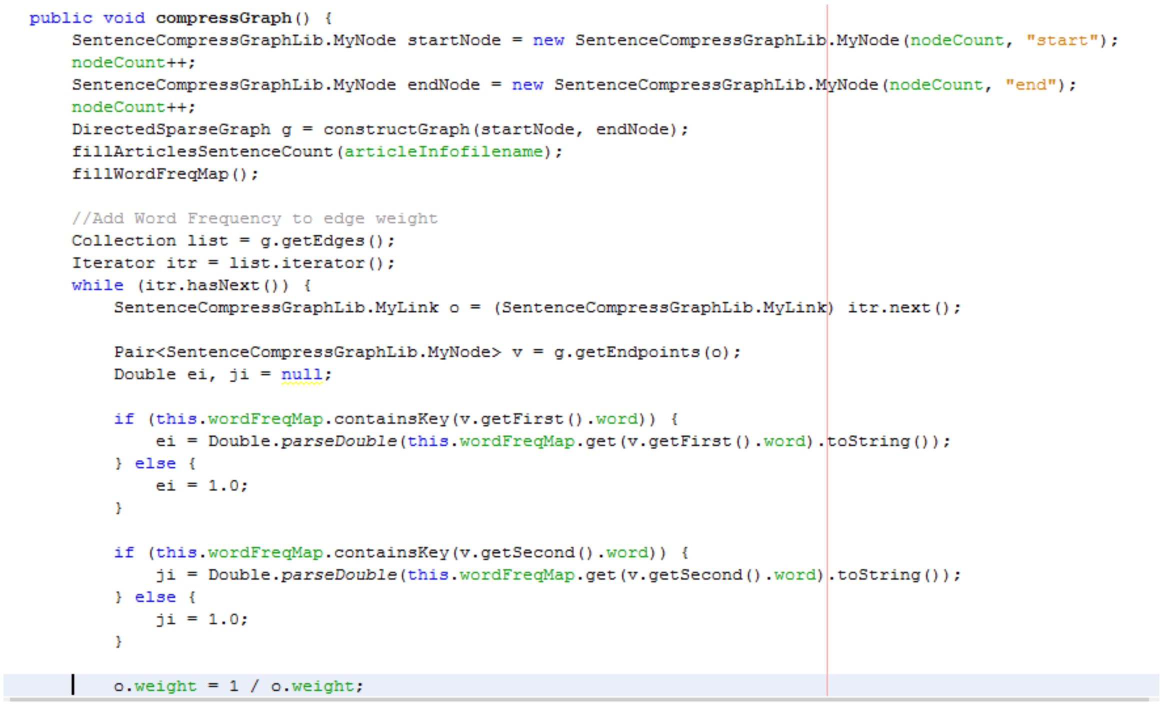


FIGURE 25 ILLUSTRATION OF COMPRESSED GRAPH USING OUR SYSTEM

As it is shown in the graph the words for “jose saramago” appearing in the four sentences has been mapped to the same nodes this is the same for the “noble prize” word. Each sentence correspond to a loop-less path from the start to the end node and the stop words are only mapped with the have same context like the stop-word “the” in phrase “the noble” it is mapped when the same words “won” and “noble” appeared in the sentences thus having same context.

Following is a snippet of the implementation of the cluster compression, it starts with construction the graph by calling function constructGraph that takes the start and end node. Then it iterate through the graph to invert the edges’ weights and divide them with the product of the frequency of the two words of their vertices.

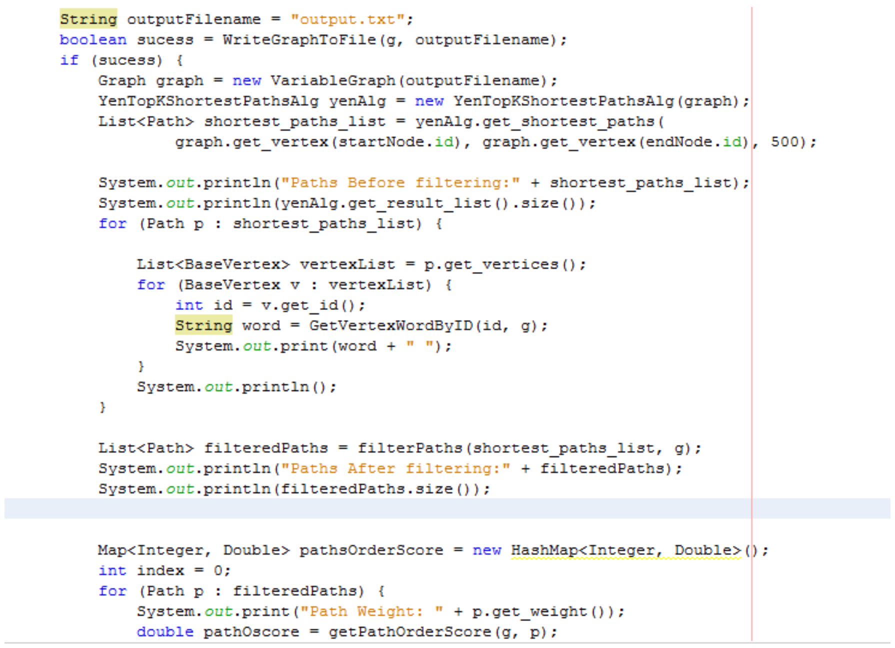


```
public void compressGraph() {
    SentenceCompressGraphLib.MyNode startNode = new SentenceCompressGraphLib.MyNode(nodeCount, "start");
    nodeCount++;
    SentenceCompressGraphLib.MyNode endNode = new SentenceCompressGraphLib.MyNode(nodeCount, "end");
    nodeCount++;
    DirectedSparseGraph g = constructGraph(startNode, endNode);
    fillArticlesSentenceCount(articleInfofilename);
    fillWordFreqMap();

    //Add Word Frequency to edge weight
    Collection list = g.getEdges();
    Iterator itr = list.iterator();
    while (itr.hasNext()) {
        SentenceCompressGraphLib.MyLink o = (SentenceCompressGraphLib.MyLink) itr.next();

        Pair<SentenceCompressGraphLib.MyNode> v = g.getEndpoints(o);
        Double ei, ji = null;

        if (this.wordFreqMap.containsKey(v.getFirst().word)) {
            ei = Double.parseDouble(this.wordFreqMap.get(v.getFirst().word).toString());
        } else {
            ei = 1.0;
        }

        if (this.wordFreqMap.containsKey(v.getSecond().word)) {
            ji = Double.parseDouble(this.wordFreqMap.get(v.getSecond().word).toString());
        } else {
            ji = 1.0;
        }

        o.weight = 1 / o.weight;
```

FIGURE 26 SNIPPET OF CODE TO COMPRESS GRAPH STARTING WITH CONSTRUCTING GRAPH AND ADJUSTING EDGES WEIGTHS

After that we use a publicly available library to execute the k-shortest paths algorithm. Following in Figure 27 is the snippet of the code where we utilize such library to find the k-shortest paths in the constructed graph. We also filter the paths based on the presence of a word with part of speech tag representing a verb. We also filter paths based on length as explained above.

The code iterate through each cluster of sentences to run the algorithm on. They loop for such iteration depends on the number of cluster being compressed. Then for each cluster the code runs the compress graph algorithm which construct graph for each cluster and uses the library for k-shortest path to find k-shortest paths (which is set to top 500 paths) and filter though each path to find the top path to represent the cluster in a compressed sentence.



```java
String outputFilename = "output.txt";
boolean sucess = WriteGraphToFile(g, outputFilename);
if (sucess) {
    Graph graph = new VariableGraph(outputFilename);
    YenTopKShortestPathsAlg yenAlg = new YenTopKShortestPathsAlg(graph);
    List<Path> shortest_paths_list = yenAlg.get_shortest_paths(
            graph.get_vertex(startNode.id), graph.get_vertex(endNode.id), 500);

    System.out.println("Paths Before filtering:" + shortest_paths_list);
    System.out.println(yenAlg.get_result_list().size());
    for (Path p : shortest_paths_list) {

        List<BaseVertex> vertexList = p.get_vertices();
        for (BaseVertex v : vertexList) {
            int id = v.get_id();
            String word = GetVertexWordByID(id, g);
            System.out.print(word + " ");
        }
        System.out.println();
    }

    List<Path> filteredPaths = filterPaths(shortest_paths_list, g);
    System.out.println("Paths After filtering:" + filteredPaths);
    System.out.println(filteredPaths.size());


    Map<Integer, Double> pathsOrderScore = new HashMap<Integer, Double>();
    int index = 0;
    for (Path p : filteredPaths) {
        System.out.print("Path Weight: " + p.get_weight());
        double pathOscore = getPathOrderScore(g, p);
```

FIGURE 27 SNIPPET OF CODE TO SEARCH AND FILTER K-SHORTEST PATHS IN THE CONSTRUCTED GRAPH

4.5. ORDER COMPRESSED SENTENCES INTO FINAL SUMMARY

The nodes in the graph encapsulate the order of the sentence it appeared in with relation to its original article. Thus, after we generate the compressed version of each cluster we sum the sentence order of each words normalized by the length of each article that word appeared in. Thus each compressed sentence will have an order score as follows:

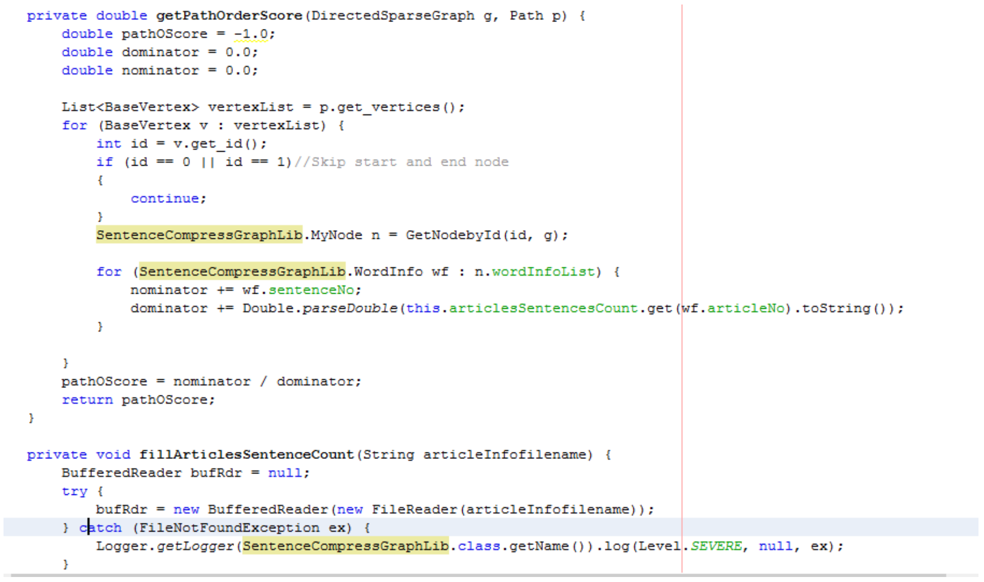
$$OrderScore(S) = \sum\nolimits_{w_i \epsilon\, S} \frac{pos_x(w_i)}{n_x}$$

$pos_x(w) is\ the\ position\ of\ sentences\ in\ which\ word\ w_i\ appears$

$n_x is\ the\ number\ of\ sentences\ in\ article\ x\ to\ which\ sentence\ of\ word\ w_i\ belong\ to$

This score gives us an indication of the order of the sentence in the flow of the information as it appeared in the original articles. We then order the sentences in an ascending fashion based on their order score. The order score ranges from just bigger than zero and up till 1. For sentence to have score of 1, all its words has to appear in the last sentences of their original documents and that shows that they would have the highest score thus will be ordered at the end of the final summary which correspond to their original order in their documents.

The following figure shows the implementation of the function that calculate the order score for the final compressed sentence.



```java
private double getPathOrderScore(DirectedSparseGraph g, Path p) {
    double pathOScore = -1.0;
    double dominator = 0.0;
    double nominator = 0.0;

    List<BaseVertex> vertexList = p.get_vertices();
    for (BaseVertex v : vertexList) {
        int id = v.get_id();
        if (id == 0 || id == 1)//Skip start and end node
        {
            continue;
        }
        SentenceCompressGraphLib.MyNode n = GetNodebyId(id, g);

        for (SentenceCompressGraphLib.WordInfo wf : n.wordInfoList) {
            nominator += wf.sentenceNo;
            dominator += Double.parseDouble(this.articlesSentencesCount.get(wf.articleNo).toString());
        }

    }
    pathOScore = nominator / dominator;
    return pathOScore;
}

private void fillArticlesSentenceCount(String articleInfofilename) {
    BufferedReader bufRdr = null;
    try {
        bufRdr = new BufferedReader(new FileReader(articleInfofilename));
    } catch (FileNotFoundException ex) {
        Logger.getLogger(SentenceCompressGraphLib.class.getName()).log(Level.SEVERE, null, ex);
    }
```

FIGURE 28 IMPLEMENTATION OF THE FUNCTION TO CALCULATE ORDER SCORE FOR COMPRESSED SENTENCE

The function main loop number of runs is determined by the number of vertices in the path and for each node it loops to sum the word's position in sentences it appeared in. This loop will run according to the number of sentences it appears in. This information is obtained from the array of word information that encapsulate the information about the word and sentences it appears in.

In the following chapter we present our system evaluation work where we explain the experiments we conducted for evaluation and discuss the results obtained.

CHAPTER 5. SYSTEM EVALUATION

In this chapter we present the experiments we conducted in order to evaluate our system. We conducted two experiments for evaluation. The first experiment aims to measure the quality of the summaries generated by our system compared to ideal human reference summaries using ROUGE evaluation system. It also compares those ROUGE scores to the scores of the summaries generated by MEAD summarization system. The second experiment uses human evaluators to evaluate the order and comprehensibility of the summaries.

5.1. DATASET

The dataset was obtained from Document Understanding Conference (DUC) for year 2004 for task 2 where newswire data is grouped into clusters of documents based on events. The news are collected from:

- AP newswire, 1998-2000
- New York Times newswire, 1998-2000
- Xinhua News Agency (English version), 1996-2000

On average, each cluster of document consists of 10 documents. We have a total of 50 of these clusters to evaluate our system and with each cluster an average of 4 references summaries are provided. Therefore, in total we will be using around 500 documents for summarization.

5.2. Baseline: MEAD summarization tool

MEAD 9 is a publicly available platform for multi-lingual summarization. It implements multiple document summarizations. MEAD v1.0 and v2.0 were developed at the University of Michigan in 2000 and early 2001. MEAD v3.01– v3.06 were written in the summer of 2001, an eight-week summer workshop on Text Summarization that was held at Johns Hopkins University [8]. It is written in Perl. We will use MEAD summarization tool as a baseline to provide baseline summaries that we can compare our work to. We chose to compare to MEAD since it provides query independent summarization as opposed to other work like MMR. The algorithm of MEAD is mentioned in details in the literature review in Chapter 2. MEAD's architecture consists of four stages. It first preprocess the cluster of documents to be summarized and converts it to MEAD's internal format which is XML-based. Then using a configuration file, it extracts features from each sentence of the cluster. Then these features are combined into a composite score for each sentence. It then further refines these scores based on dependencies with other sentences like information redundancy or ordering. MEAD orders the sentences using the same order of sentences as they appear in the original document, with documents ordered chronologically. It provide by default a compression ratio of 20% of the number of sentences in the whole cluster of documents to be summarized.

5.3. Evaluation Experiments

We ran two different experiments for the final system evaluation. The first experiment aimed to provide ROUGE score to evaluate the quality of our summaries. The second experiment used human evaluators to evaluate the ordering of the sentences in the summaries.

[9] MEAD http://www.summarization.com/mead/

5.3.1. Experiment One: ROUGE score for Automatically Generated summaries

Objective: The objective of this experiment is to measure the quality of the summaries generated by our system using the dataset compared to the provided ideal human reference summaries. We use the ROUGE evaluation system which compare the automated summaries to that of their human ideal counterparts. We also calculate the ROUGE score of the summaries generates by MEAD summarization system to act as a baseline for our system.

Method: We ran our system to provide the summary for 50 clusters of documents. Each cluster constitutes on average 10 documents. We used publicly available scripts in order to convert the automated summaries as well as ideal human reference summaries to ROUGE format. Then we ran the ROUGE evaluation system to measure the ROUGE-1, ROUGE-2, and ROUGE-S4 for the 50 generated summaries in comparison to the human reference summaries. We also ran the same dataset using MEAD system to produced 50 summaries to be used as a baseline to compare our system to. We processed the MEAD summaries to convert it to ROUGE format and then ran the ROUGE system to measure the ROUGE-1, ROUGE-2 and ROUGE-S4. For each of those measures we obtained the recall, precision and f-score.

Results: The following table shows a comparison between the recall, precision and f-score for the ROUGE-1, ROUGE-2 and ROUGE-S4 measures of our system summaries and the recall, precision and f-score for the same measures of the MEAD system summaries.

TABLE 13 COMPARISON BETWEEN ROUGE SCORES OF OUR SYSTEM AND THAT OF MEAD

	Recall		Precision		F-score	
Measure	Our System	MEAD	Our System	MEAD	Our System	MEAD
ROUGE-1	0.49157	**0.71229**	**0.16136**	0.0692	**0.23736**	0.12454
ROUGE-2	0.11616	**0.20321**	**0.03868**	0.01923	**0.05669**	0.03469
ROUGE-SU4	0.21252	**0.47755**	**0.0249**	0.00508	**0.04251**	0.01

Discussion:

Our system provided better precision and f-score. On average our system has a percentage increase of 2% for precision and 1.6% increase in f-score than those of MEAD while MEAD has an increase of 0.8% in recall. In addition, MEAD system provided a less compressed version of the documents while our system provided more compressed versions. For example, in one dataset of 10 documents each with an average of 19 sentences and ranges from 10-28 sentences, our summary consisted of 14 sentences only while MEAD's summary consisted of 38 sentences. This contributed to their recall scores being higher than ours since their summaries are longer and contains more material that would match the ones in the references summaries. We include in our appendix A the full text for a sample of 8 documents to be summarized in one summary along with 4 human references that act as model references. In addition, we include our generated summary for those documents and the summary generated by MEAD system. The 8 articles are from the AP newswire discussing a fire that burned down a dance hall in Sweden during one Halloween night. The articles lengths ranges from 36 to 45 sentences. The average length of the articles was 40 sentences. Our generated summary was 14 sentences long as opposed to 68 sentences long summary generated by MEAD.

5.3.2. EXPERIMENT TWO: ORDER OF SENTENCES IN SUMMARIES AND THEIR COMPREHENSIBILITY

Objective: The objective of this experiment is to evaluate the order score we proposed in our system. Since that ROUGE only measures word overlap the order of the summary is not captured in this measurement. Thus we utilized human evaluator to read a sample subset of the summaries and score them for order and comprehensibility. Relaying on human evaluator made us select a sample of the dataset as opposed to the whole dataset. So we selected 30 % of the summaries which amounted to 15 summaries, selected at random.

Method: In this experiment, different human evaluators were employed to evaluate the order of sentences in the final summaries produced by the system and how well it is a comprehendible text. The evaluators were provided with 15 summaries from the generated set selected at random. They then rated the final summary based on three ratings: 0- mainly unordered sentences that does not convey any meaning, 1- somewhat ordered summary but still understandable, 2- perfect or near perfect order summary which they understood. We then averaged the scores given to the summaries to have an indication of the method used to order the final summary and how coherent is the flow of information in our system generated summaries. Evaluators were also asked to count the number of sentences that are totally ungrammatical and incomprehensible for each of the summaries they evaluated.

Results: The following table shows the percentage of each score given by each of the three judges and the average of their scores.

TABLE 14 PERCENTAGE OF EACH SCORE GIVEN BY THREE JUDGES EVALUATING THE ORDER OF GENERATED SUMMARIES

Score	Judge A (%)	Judge B (%)	Judge C (%)
0	0	0	0
1	18.75	25	37.5
2	81.25	75	62.5

In addition, the following table shows the percentage of sentences marked by each evaluator.

TABLE 15 EVALUATORS PERCENTAGE FOR UNGRAMMATICAL SENTENCES

Evaluator	Percentage
A	13.61
B	8.23
C	8.93

Discussion:

These results show that the majority of the summaries falls in the 2 and1 score with none of the evaluated summaries being incomprehensible. On average the score 2 summaries constitute 72% of the evaluated summaries. This shows that the order criteria we proposed produces comprehensible summaries. Evaluators were also asked to count the number of sentences that are totally ungrammatical and incomprehensible for each of the summaries they evaluated. We calculated the percentage of such sentences in regard to the total number of sentences in each of the summaries On average the percentage of those incomprehensible sentence where 10.9% of the sentences of the summaries. We ran a separate experiment in order to look in to the order of the final summary. Since that the final sentences is a compressed version of sentences from the different documents we do not have the exact order of the sentence in one document. Thus utilized human evaluators to read generated summaries and evaluate them based on the order and comprehensibility. This further validate our system where we human evaluators indicate to what degree they understood the summary and how ordered did it appear to them. Since that ROUGE score only measures overlapping in the words of the generated summaries and those in ideal references, it does not indicate the order of the summary. Having human evaluators inspect the summaries validate the order of the generated summaries as well as validate the semantics of it. In the next chapter we present our conclusion for our thesis work along with our main contribution in addition to our future work.

Chapter 6. Conclusion and Future Work

6.1. Conclusion

In our work, we tackled the problem of automatic multiple document summarization with the aim to investigate various aspects of the summarization process. We started with reviewing the literature and worked on investigating various research problems. Our first question dealt with scoring sentences in a document for significance. We tried various features and combinations of features. Through empirical results, we came to the conclusion that scoring sentences based on the presence of the document high frequent words as well as words appearing in first and last sentences yielded the best results in the features configurations we tested. Secondly, we looked into selecting a similarity measure to select sentences. We tried two approaches, the first uses word frequency vector and the second uses word semantic similarity measure. Throughout our experiments, we found that the use of the word frequency vector was a better way to provide clusters of similar sentences. We then investigated a graph-based representation of text that is used to compress sentences based on [1] and [2]. Throughout, that approach we were able to compress similar sentences into clusters that are comprehensible. We then moved on to design a system for multiple document summarization that uses extractive methods as well as “shallow” abstractive methods. The system starts by selecting significant sentences from each of the documents in the set and then clustering selected sentences into clusters based on similarity. It then processes each cluster constructing a graph-based representation of the sentences and using that graph to compress sentences into one sentence. This is accomplished by finding the shortest path from start to end node and where the edges’ weights are calculated in order to boost paths that go through the high frequent terms. Finally, we ordered the compressed sentences into a concise, well-formed summary. Our results, as compared to MEAD systems, are promising. In addition, summaries produced by our system are much shorter in length than that produced by MEAD system. Finally, evaluators of the order of the final summaries found that the majority of the summaries they tested where in good order.

Main Contribution: Our main contribution stems from our use of the graph-based representation of text for sentence compression in a full multiple document summarization system. We also introduced a new way to order sentences in final summary using such representation. In addition, we tackled various research questions in automatic summarization and tested the word semantic similarity measure on English language where it was used on Chinese language in the literature. We believe our approach provide a way to provide a "shallow" abstractive method to summarization that do not require intensive NLP or prior knowledge. Our final summaries are more concise than their MEAD system counterparts. The use of such an approach in full multiple document summarization to the best of our knowledge has not been tested before.

6.2. FUTURE WORK

Using our framework we aim to test the approach on other languages like Arabic. This requires only the change of the part of speech tagger to use an Arabic language one. We also aim to test it on other genres of documents -not only news- to test how well it scales. We intend to work on improving the sentence compression to provide better quality sentences that are more concise and as grammatically perfect as possible. We want to consider more filtering techniques to make sure the final summary is grammatically sound. This might include mapping the sentences to some Part of Speech constraints. We will also investigate other means for ordering final summary. We are also working on publishing our findings.

References

[1] K. Filippova, "Multi-sentence compression: Finding shortest paths in word graphs," in *Proceedings of the 23rd International Conference on Computational Linguistics*, Beijing, China, 2010.

[2] K. Ganesan, C. Zhai and J. Han, "Opinosis : A Graph-Based Approach to Abstractive Summarization of Highly Redundant Opinions," in *Proceedings of the 23rd International Conference on Computational Linguistics*, Beijing, China, 2010.

[3] K. Ramanathan, Y. Sankarasubramaniam, N. Mathur and A. Gupta, "Document summarization using Wikipedia," in *Proceedings of the First International Conference on Intelligent Human Computer Interaction*, Allahabad, India, 2009.

[4] I. Mani and M. T. Maybury, Advances in Automatic Text Summarization, The MIT Press, July 1999.

[5] Inderjeet Mani, "Summarization Evaluation : An Overview," in *Proceedings of the Second NTCIR Workshop on Research in Chinese & Japanese Text Retrieval and Text Summarization*, Tokyo, Japan, 2001.

[6] U. Hahn and I. Mani, "The Challanges of Automatic Summarization," *Computer,* vol. 33, no. 11, pp. 29 - 36, 2000.

[7] Z. Pei-ying and L. Cun-he, "Automatic Text Summarization Based on Sentences Clustering and Extraction," in *Proceedings of IEEE International Conference on Computer Science and Information Technology (ICCSIT)*, Beijing, 2009.

[8] D. R. Radev, H. Jing and M. Budzikowska, "Centroid-based summarization of multiple documents: sentence extraction, utility-based evaluation, and user studies," in *Proceedings of the 2000 NAACL-ANLP Workshop on Automatic summarization*, PA, USA, 2000.

[9] U. Hahn and I. Mani, *Figure 1. Architecture for extraction (knowledge-poor summarization) Taken from The Challenges of Automatic Summarization,* vol. 33, Computer, 2000, p. 30.

[10] U. Hahn and I. Mani, *Figure 2. A classifier that learns how to summarize Taken from The Challanges of Automatic Summarization,* vol. 33, Computer, 2000, p. 31.

[11] U. Hahn and I. Mani, *Figure 3. Architecture for abstraction (knowledge-rich summarization) Taken from The Challanges of Automatic Summarization,* vol. 33, Computer, 2000, p. 32.

[12] H. Luhn, "The automatic creation of literature abstracts," *IBM Journal of research and development,* vol. 2, no. 2, p. 159–165, 1958.

[13] H. P. Edmundson, "New Methods in Automatic Extracting," *Journal of the ACM,* vol. 16, no. 2, p. 264–285, April 1969.

[14] J. Kupiec, J. Pedersen and F. Chen, "A trainable document summarizer," in *Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval*, NY,USA, 1995.

[15] S. H. Myaeng and D. Jang, "Development and Evaluation of a Statistically-Based Document Summarization System," in *Advances in automatic text summarization*, The MIT Press, 1999, p. 61–70.

[16] K. Svore, L. Vanderwende and C. Burges, "Enhancing Single-document Summarization by Combining RankNet and Third-party Sources," in *In Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, Prague, Czech Republic, 2007.

[17] R. Barzilay and M. Elhadad, "Using Lexical Chains for Text Summarization," in *In Proceedings of the ACL Workshop on Intelligent Scalable Text Summarization*, 1997.

[18] J. Morris and G. Hirst, "Lexical Cohesion Computed by Thesaural Relations as an Indicator of the Structure of Text," *Computational Linguistics,* vol. 17, no. 1, pp. 21-48, 1991.

[19] R. Barzilay and M. Elhadad, *Figure 3 Step 3 Interpretation 1 Taken from Using Lexical Chains for Text Summarization,* In Proceedings of the ACL Workshop on Intelligent Scalable Text Summarization, 1997, p. 14.

[20] D. Marcu, "Discourse trees are good indicators of importance in text," in *Advances in Automatic Text Summarization*, The MIT Press, 1999, pp. 123-136.

[21] K. Ono, K. Sumita and S. Miike, "Abstract Generation based on rhetorical structure extraction," in *Proceedings of the 15th conference on Computational linguistics*, 1994.

[22] A. F. T. M. Dipanjan Das, *A Survey on Automatic Text Summarization,* 2007.

[23] J. G. V. Mittal, J. Goldstein, V. Mittal, J. Carbonell and M. Kantrowitz, "Multi-Document Summarization By Sentence Extraction," in *In Proceedings of the ANLP/NAACL Workshop on Automatic Summarization*, 2000.

[24] J. G. V. Mittal, J. Goldstein, V. Mittal, J. Carbonell and M. Kantrowitz, *Figure 1: Definition of multi-document summarization algorithm - MMR-MD Taken From Multi-Document*

Summarization By Sentence Extraction, In Proceedings of the ANLP/NAACL Workshop on Automatic Summarization, 2000, p. 44.

[25] A. L. Berger and V. O. Mittal, "OCELOT: a system for summarizing Web pages," in *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, NY, USA, 2000.

[26] A. L. Berger and V. O. Mittal, *Algorithm 3: Readable gisting Taken from OCELOT: a system for summarizing Web pages,* Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval, 2000, p. 146.

[27] A. L. Berger and V. O. Mittal, *Figure 5: Selected output from OCELOT Taken from OCELOT: a system for summarizing Web pages,* Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 2000, p. 150.

[28] J.-t. Sun, Q. Yang and Y. Lu, "Web-page summarization using clickthrough data," in *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2005.

[29] Y. Gong and X. Liu, "Generic text summarization using relevance measure and latent semantic analysis," in *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, 2001.

[30] E. Amitay and C. Paris, "Automatically summarising web sites: is there a way around it?," in *Automatically summarising web sites: is there a way around it?*, 2000.

[31] C. P. E. Amitay, *Fig. 2. Relations between documents as detected by the InCommonSense system Taken from Automatically summarising web sites: is there a way around it?,* Proceedings of the ninth international conference on Infromation and Knowledge Managment, 2000, p. 175.

[32] E. Filatova, "Multilingual Wikipedia , Summarization , and Information Trustworthiness," in *Proceedings of the SIGIR Workshop on Information Access in a Multilingual World*, Boston, Massachusetts, 2009.

[33] A. Nenkova, R. Passonneau and K. McKeown, "The Pyramid Method," in *ACM Transactions on Speech and Language Processing*, 2007.

[34] E. Filatova, *Table 1: Algorithm outline Taken from Multilingual Wikipedia , Summarization , and Information Trustworthiness,* Boston, Massachusetts: Proceedings of the SIGIR Workshop on Information Access in a Multilingual World, 2009.

[35] E. Filatova, *Table 2: Three-level summaries for Gene Autry Taken from Multilingual Wikipedia , Summarization , and Information Trustworthiness,* Boston, Massachusetts: Proceedings of the SIGIR Workshop on Information Access in a Multilingual World, 2009.

[36] E. Filatova, *Figure 1: Combined results Taken from Multilingual Wikipedia , Summarization , and Information Trustworthiness,* Boston, Massachusetts: Proceedings of the SIGIR Workshop on Information Access in a Multilingual World, 2009.

[37] K. Ganesan, C. Zhai and J. Han, *Figure 1: Sample Opinosis-Graph Taken from Opinosis : A Graph-Based Approach to Abstractive Summarization of Highly Redundant Opinions,* PA, USA: Proceedings of the 23rd International Conference on Computational Linguistics, 2010, p. 342.

[38] K. Filippova, *Figure 1: Word graph generated from sentences (1-4) and a possible compression path Taken from Multi-sentence compression: Finding shortest paths in word graphs,* roceedings of the 23rd International Conference on Computational Linguistics, 2010.

[39] L. Plaza, E. Lloret and A. Aker, "Improving automatic image captioning using text summarization techniques," in *Proceedings of the 13th international conference on Text, speech and dialogue*, Berlin, Heidelberg, 2010.

[40] L. Morales, "Concept-graph based biomedical automatic summarization using ontologies," in *Proceedings of the 3rd Textgraphs Workshop on Graph-Based Algorithms for Natural Language Processing*, PA, USA, 2008.

[41] E. Lloret and M. Palomar, "A gradual combination of features for building automatic summarisation systems," in *Proceedings of the 12th International Conference on Text, Speech and Dialogue*, Berlin, Heidelberg, 2009.

[42] D. E. Liddy, "Discourse Level Structure," in *Information Processing and Managment*, 1991.

[43] C.-y. Lin, "Rouge: A package for automatic evaluation of summaries," in *Proc. ACL workshop on Text Summarization Branches Out*, 2004.

[44] W. Visser and M. Wieling, "Sentence-based Summarization of Scientific Documents.The design and implementation of an online available automatic summarizer.," 2007. Report, last retrieved February 2014 from http://www.martijnwieling.nl/files/wielingvisser05automaticsummarization.pdf

APPENDIX A. SAMPLE SUMMARY

Following is a sample summary of 8 articles of AP newswire discussing a fire that burned down a dance hall in Sweden during one Halloween night. The articles lengths ranges from 36 to 45 sentences. The average length of the articles was 40 sentences. Our generated summary was 14 sentences long as opposed to 68 sentences long summary generated by MEAD.

Following is the text of the original articles:

Article 1

A fire turned a dance hall jammed with teen-age Halloween revelers into a deathtrap, killing 60 people and injuring 162 others in Sweden's second-largest city.
Police earlier had reported 65 dead, but backed off that figure Friday evening.
''The earlier information that police gave out was wrong,'' Hand Carlsson, the lead detective in the case, told a news conference.
The fast-spreading fire that broke out just a few minutes before midnight Thursday gutted the building and left rescuers facing a hideous scene that local rescue service leader Lennart Olin likened to a ``gas chamber.''
The cause of the fire had not been determined as of Friday evening.
Although an estimated 400 people, most aged 13 to 18, were at the dance on the building's second floor, the facility had approval for a maximum capacity of 150, said Carlsson.
The fire, at the facilities of the Macedonian Association local immigrant group, was the deadliest in modern Swedish history.
In 1978, 20 people died in a fire at a hotel in the town of Boraas.
Police said most victims choked to death on smoke and poisonous gases; 59 bodies were found at the scene and one other died later.
Of the injured, at least 57 were in intensive care, according to Sven Martinell, spokesman for the local medical authorities.
The building did not have sprinklers and was not required to have them, officials said.
The dance was attended mostly by immigrants or children of immigrants.
Police said the dead or injured represented at least 19 nationalities, including Somalis, Ethiopians, Iraqis, Iranians and Swedes, along with people from current and former Yugoslavia and unspecified Latin American countries.
The Macedonian Association, which rents space in the building, had in turn rented the facility out to others for the dance, Carlsson said.
That person was not immdiately identified.
Binan Atta was walking to the Macedonian Association when he saw the fire.
He said he raced in and pulled to safety several people, including a friend.
``His clothes had burned off.
His skin was red and bubbly,'' Atta said.
``Lots of kids were just screaming,'' he added at Hammarkullen Lutheran Church, where several dozen family and friends of victims gathered.

``I saw about 10 people in windows who just jumped.
They didn't even look down'' beforehand.
Fire officials were alerted at 11:43 p.m. (2243 GMT) Thursday and had a fire truck at the scene four minutes later, rescue workers said.
The blaze was already consuming the building.
The building had just two exits, one of which was blocked by fire, city police technician Stephen Holmberg was quoted as saying by the Swedish news agency TT.
``It was a panic,'' Goteborg police spokesman Bengt Staaf said, with youths trampling each other to get out, and other youths scuffling with police to get in and attempt to help friends.
Olin said there were indications that the fire could have been set.
``The fact that it spread so fast indicates that it was not a normal fire,'' he said.
Goteborg is about 500 kilometers (300 miles) southwest of the capital, Stockholm.
The crowd in the second story of the building contained mostly 13 to 18 year olds celebrating Halloween and a holiday weekend.
``A night full of expectation, happiness over extra leave from school and high spirits in anticipation of a weekend was brutally and suddenly changed into a tragedy of incomprehensible dimensions,'' the Goteborg city council said in a statement.
``Goteborg is today a city in shock.''
Prime Minister Goeran Persson visited the fire site at midday and King Carl XVI Gustav made a statement of condolence.
Jamal Fawz, 15, told TT that he was out on the dance floor when the blaze started with about 400 people inside.
``It looked like it started in the ceiling, and lamps and loudspeakers fell to the floor,'' he was quoted as saying.
``It was chaos.
Everybody was trying to get out and people trampled on each other on the way to the exit. ...
Others kicked out the windows and jumped out.''
Ambulances were called in from several nearby communities.
The Goteborg rescue services also brought city buses into service to help transport the injured.
Anna-Lisa Saar, a social worker at Oestra Hospital, where many of the victims were taken, said identifying many of them was difficult.
``Maybe you have teen-agers yourself and know how they are ...
They maybe don't have their own identification, but have that of a friend who is a year older.
Girls don't carry their identification on them, but in a bag and maybe that wasn't lying with the body,'' she said, according to TT.
Goteborg has about 435,000 people.

Article 2

A fire turned a Swedish dance hall jammed with teen-age Halloween revelers into a deathtrap, killing at least 60 people and injuring about 180.
The fast-spreading fire completely gutted the two-story building and left rescuers facing a hideous scene that local rescue service leader Lennart Olin likened to a ``gas chamber.''
Although an estimated 400 people, most aged 13 to 18, were at the dance on the upper floor, the facility had approval for a maximum capacity of 150, Hans Carlsson, the detective leading the disaster investigation.
The fire, at the facilities of the Macedonian Association local immigrant group, was the deadliest in modern Swedish history since 1978, when 20 people died in the town of Boraas.
On Friday, police said most victims choked to death on smoke and poisonous gases; 59 bodies were found at the scene, and the 60th victim died at a hospital, officials said.
The injured included about 20 in serious condition.

The dance apparently was attended mostly by immigrants or children of immigrants.
Police said the list of injured included Somalis and people from current and former Yugoslavia.
The Macedonian Association had rented the facility out for the dance, Carlsson said.
That person was not immediately identified.
Binan Atta was walking to the Macedonian Association when he saw the fire.
He said he raced in and pulled to safety several people, including a friend.
``His clothes had burned off.
His skin was red and bubbly,'' Atta said.
``Lots of kids were just screaming,'' he added at Hammarkullen Lutheran Church, where several dozen family and friends of victims gathered.
``I saw about 10 people in windows who just jumped.
They didn't even look down'' beforehand.
Fire officials were alerted at 11:43 p.m. (2243 GMT) Thursday and had a fire truck at the scene four minutes later, rescue workers said.
The blaze was already consuming the brick building.
About 300 or 400 people were inside, police said.
Many escaped on their own.
Police rescued 40.
The Swedish news agency TT said the association had permission to have only 150 people in the facility.
The building had just two exits, one of which was blocked by fire, city police technician Stephen Holmberg was quoted as saying by TT.
``It was a panic,'' Goteborg police spokesman Bengt Staaf said, with youths trampling each other to get out, and other youths scuffling with police to get in and attempt to help friends.
The cause of the fire was not immediately known.
``What we know is that there was an explosion,'' Edmundson said.
Olin said there were indications that the fire could have been set.
``The fact that it spread so fast indicates that it was not a normal fire,'' he said.
About 180 people were taken to hospitals.
Seven of the most severely injured were taken by helicopter to burn clinics in other cities, the report said.
Goteborg is about 500 kilometers (300 miles) southwest of the capital, Stockholm.
The dance was filled with teen-agers celebrating Halloween and a holiday weekend.
``A night full of expectation, happiness over extra leave from school and high spirits in anticipation of a weekend was brutally and suddenly changed into a tragedy of incomprehensible dimensions,'' the Goteborg city council said in a statement.
``Goteborg is today a city in shock.''
Prime Minister Goeran Persson visited the fire site at midday and King Carl XVI Gustav made a statement of condolence.

Article 3

A fire turned a dance hall jammed with teen-age Halloween revelers into a deathtrap, killing at least 60 people and injuring about 180 in Sweden's second-largest city.
The fast-spreading fire completely gutted the building and left rescuers facing a hideous scene that local rescue service leader Lennart Olin likened to a ``gas chamber.''
Although an estimated 400 people, most aged 13 to 18, were at the dance on the building's second floor, the facility had approval for a maximum capacity of 150, said Hans Carlsson, the detective leading the disaster investigation.
The fire, at the facilities of the Macedonian Association local immigrant group, was the deadliest in modern Swedish history.

In 1978, 20 people died in a fire at a hotel in the town of Boraas.
At a news conference Friday morning, police said most victims choked to death on smoke and poisonous gases; 59 bodies were found at the scene and the 60th victim died at a hospital, officials said.
The dance apparently was attended mostly by immigrants or children of immigrant.
Police said the list of injured included Somalis and people from current and former Yugoslavia.
The Macedonian Association, which rents space in the building, had in turn rented the facility out to another person for the dance, Carlsson said.
That person was not immdiately identified.
Binan Atta was walking to the Macedonian Association when he saw the fire.
He said he raced in and pulled to safety several people, including a friend.
``His clothes had burned off.
His skin was red and bubbly,'' Atta said.
``Lots of kids were just screaming,'' he added at Hammarkullen Lutheran Church, where several dozen family and friends of victims gathered.
``I saw about 10 people in windows who just jumped.
They didn't even look down'' beforehand.
Fire officials were alerted at 11:43 p.m. (2243 GMT) Thursday and had a fire truck at the scene four minutes later, rescue workers said.
The blaze was already consuming the building.
The building had just two exits, one of which was blocked by fire, city police technician Stephen Holmberg was quoted as saying by the Swedish news agency TT.
``It was a panic,'' Goteborg police spokesman Bengt Staaf said, with youths trampling each other to get out, and other youths scuffling with police to get in and attempt to help friends.
The cause of the fire was not immediately known, although officials said it apparently began with an explosion.
Olin said there were indications that the fire could have been set.
``The fact that it spread so fast indicates that it was not a normal fire,'' he said.
About 180 people were taken to hospitals with injuries, and about 20 were in intensive care.
Seven of the most severely injured were taken by helicopter to burn clinics in other cities.
Goteborg is about 500 kilometers (300 miles) southwest of the capital, Stockholm.
The crowd in the second story of the building contained mostly 13 to 18 year olds celebrating Halloween and a holiday weekend.
``A night full of expectation, happiness over extra leave from school and high spirits in anticipation of a weekend was brutally and suddenly changed into a tragedy of incomprehensible dimensions,'' the Goteborg city council said in a statement.
``Goteborg is today a city in shock.''
Prime Minister Goeran Persson visited the fire site at midday and King Carl XVI Gustav made a statement of condolence.
Jamal Fawz, 15, told TT that he was out on the dance floor when the blaze started with about 400 people inside.
``It looked like it started in the ceiling, and lamps and loudspeakers fell to the floor,'' he was quoted as saying.
``It was chaos.
Everybody was trying to get out and people trampled on each other on the way to the exit. ... Others kicked out the windows and jumped out.''
Ambulances were called in from several nearby communities.
The Goteborg rescue services also brought city buses into service to help transport the injured.
Anna-Lisa Saar, a social worker at Oestra Hospital, where many of the victims were taken, said identifying many of them was difficult.

``Maybe you have teen-agers yourself and know how they are ...
They maybe don't have their own identification, but have that of a friend who is a year older.
Girls don't carry their identification on them, but in a bag and maybe that wasn't lying with the body,'' she said, according to TT.
Goteborg has about 435,000 people.

Article 4

A fire turned a Swedish dance hall jammed with teen-age Halloween revelers into a deathtrap, killing at least 60 people and injuring about 180.
``It reminded me of the gas chambers at Auschwitz,'' local rescue service leader Lennart Olin said on national radio, describing the sight when rescuers first entered the building in Goteborg, a city of half a million people on the country's west coast.
The fire was the deadliest in modern Swedish history since 1978, when 20 people died in the town of Boraas.
``We are still searching the building ... but so far we have found 60 dead,'' Goteborg police official Jan Edmundson said on national radio.
At a news conference Friday morning, police said most victims choked to death on smoke and poisonous gases.
The injured toll was lowered slightly to 180, but included seven people who needed treatment at emergency burn centers.
Binan Atta was walking to the Macedonian Association when he saw the fire.
He said he raced in and pulled to safety several people, including a friend.
``His clothes had burned off.
His skin was red and bubbly,'' Atta said.
``Lots of kids were just screaming,'' he added at Hammarkullen Lutheran Church, where several dozen family and friends of victims gathered.
``I saw about 10 people in windows who just jumped.
They didn't even look down'' beforehand.
Fire officials were alerted at 11:43 p.m. (2243 GMT) Thursday and had a fire truck at the scene four minutes later, rescue workers said.
The blaze was already consuming the two-story brick building.
The dance was on the second floor.
About 300 or 400 people were inside, police said.
Many escaped on their own.
Police rescued 40, and recovered 59 bodies.
The 60th victim died at a hospital.
``It was a panic,'' Goteborg police spokesman Bengt Staaf said, with youths trampling each other to get out, and other youths scuffling with police to get in and attempt to help friends.
The cause of the fire was not immediately known.
``What we know is that there was an explosion,'' Edmundson said.
Olin said there were signs that the fire was set.
``The fact that it spread so fast indicates that it was not a normal fire,'' he said.
The Swedish news agency TT reported 190 people were taken to hospitals with injuries, and about 20 were in intensive care.
Seven of the most severely injured were taken by helicopter to burn clinics in other cities, the report said.
Goteborg is about 500 kilometers (300 miles) southwest of the capital, Stockholm.
The fire broke out about midnight (2300 GMT Thursday) in building of the local Macedonian Association, which organized the dance.

The crowd in the second story of the building contained mostly 13 to 18 year olds celebrating Halloween and a holiday weekend.
``A night full of expectation, happiness over extra leave from school and high spirits in anticipation of a weekend was brutally and suddenly changed into a tragedy of incomprehensible dimensions,'' the Goteborg city council said in a statement.
``Goteborg is today a city in shock.''
Jamal Fawz, 15, told TT that he was out on the dance floor when the blaze started with about 400 people inside.
``It looked like it started in the ceiling, and lamps and loudspeakers fell to the floor,'' he was quoted as saying.
``It was chaos.
Everybody was trying to get out and people trampled on each other on the way to the exit. ... Others kicked out the windows and jumped out.''
Ambulances were called in from several nearby communities.
The Goteborg rescue services also brought city buses into service to help transport the injured.
Olin said the rescue service inspected the building in April 1997 and ``fulfilled all possible demands as far as emergency exits and the possibility for fast evacuation.''
Anna-Lisa Saar, a social worker at Oestra Hospital, where many of the victims were taken, said identifying many of them was difficult.
``Maybe you have teen-agers yourself and know how they are ...
They maybe don't have their own identification, but have that of a friend who is a year older.
Girls don't carry their identification on them, but in a bag and maybe that wasn't lying with the body,'' she said, according to TT.
*Goteborg has about 435,000 peopl*e**.

Article 5

A fire turned a dance hall jammed with teen-age Halloween revelers into a deathtrap, killing 65 people and injuring 157 others in Sweden's second-largest city.
The fast-spreading fire that broke out just a few minutes before midnight Thursday gutted the building and left rescuers facing a hideous scene that local rescue service leader Lennart Olin likened to a ``gas chamber.''
The cause of the fire had not been determined as of Friday evening.
Although an estimated 400 people, most aged 13 to 18, were at the dance on the building's second floor, the facility had approval for a maximum capacity of 150, said Hans Carlsson, the detective leading the disaster investigation.
The fire, at the facilities of the Macedonian Association local immigrant group, was the deadliest in modern Swedish history.
In 1978, 20 people died in a fire at a hotel in the town of Boraas.
Police said most victims choked to death on smoke and poisonous gases; 59 bodies were found at the scene and six others died later.
Of the injured, 57 were in intensive care, said Sven Martinell, spokesman for the local medical authorities.
The building did not have sprinklers and was not required to have them, officials said.
The dance was attended mostly by immigrants or children of immigrants.
Police said the dead or injured represented at least 19 nationalities, including Somalis, Ethiopians, Iraqis, Iranians and Swedes, along with people from current and former Yugoslavia and unspecified Latin American countries.
The Macedonian Association, which rents space in the building, had in turn rented the facility out to others for the dance, Carlsson said.
That person was not immdiately identified.

Binan Atta was walking to the Macedonian Association when he saw the fire.
He said he raced in and pulled to safety several people, including a friend.
``His clothes had burned off.
His skin was red and bubbly,'' Atta said.
``Lots of kids were just screaming,'' he added at Hammarkullen Lutheran Church, where several dozen family and friends of victims gathered.
``I saw about 10 people in windows who just jumped.
They didn't even look down'' beforehand.
Fire officials were alerted at 11:43 p.m. (2243 GMT) Thursday and had a fire truck at the scene four minutes later, rescue workers said.
The blaze was already consuming the building.
The building had just two exits, one of which was blocked by fire, city police technician Stephen Holmberg was quoted as saying by the Swedish news agency TT.
``It was a panic,'' Goteborg police spokesman Bengt Staaf said, with youths trampling each other to get out, and other youths scuffling with police to get in and attempt to help friends.
Olin said there were indications that the fire could have been set.
``The fact that it spread so fast indicates that it was not a normal fire,'' he said.
Goteborg is about 500 kilometers (300 miles) southwest of the capital, Stockholm.
The crowd in the second story of the building contained mostly 13 to 18 year olds celebrating Halloween and a holiday weekend.
``A night full of expectation, happiness over extra leave from school and high spirits in anticipation of a weekend was brutally and suddenly changed into a tragedy of incomprehensible dimensions,'' the Goteborg city council said in a statement.
``Goteborg is today a city in shock.''
Prime Minister Goeran Persson visited the fire site at midday and King Carl XVI Gustav made a statement of condolence.
Jamal Fawz, 15, told TT that he was out on the dance floor when the blaze started with about 400 people inside.
``It looked like it started in the ceiling, and lamps and loudspeakers fell to the floor,'' he was quoted as saying.
``It was chaos.
Everybody was trying to get out and people trampled on each other on the way to the exit. ...
Others kicked out the windows and jumped out.''
Ambulances were called in from several nearby communities.
The Goteborg rescue services also brought city buses into service to help transport the injured.
Anna-Lisa Saar, a social worker at Oestra Hospital, where many of the victims were taken, said identifying many of them was difficult.
``Maybe you have teen-agers yourself and know how they are ...
They maybe don't have their own identification, but have that of a friend who is a year older.
Girls don't carry their identification on them, but in a bag and maybe that wasn't lying with the body,'' she said, according to TT.
Goteborg has about 435,000 people.

Article 6

Forensic experts examining heavily burned bodies were able Saturday to identify more of the 60 young people who died in a dance hall fire, but the catastrophe's most tormenting question was still unanswered.
How could it happen; what caused the flames that raced through a hall packed far beyond capacity, blocking one of the exits and forcing panicked teen-agers to flee down the one remaining staircase and leap out of second-story windows?

''As long as the technicians haven't established the cause of the fire, we don't know if it's arson or not,'' Goteborg chief prosecutor Ulf Noren said Saturday evening.
Earlier in the day, Noren had said it was a ''50-50'' chance that the fire was arson, prompting wide speculation that authorities had tracked down new clues.
But Noren later retracted the remark, saying he'd meant only that no possibilities were being excluded.
As investigators worked to find the cause, examiners identified another 22 of the bodies, bringing the total to 40, and officials said 49 people were released from hospital.
Of the 162 people who suffered non-fatal injuries in the Thursday night fire, 76 remain hospitalized.
Most of the victims were immigrants or of immigrant parentage, from countries including Iraq, Iran, Somalia, Ethiopia and current and former Yugoslavia.
The first call alerting authorities to the fire was made in heavily accented Swedish and that, combined with noise and the caller's distress, delayed the fire squads' response by several minutes.
Per-Olof Ortarsen of Goteborg's emergency services line said the call was so hard to understand that it took three minutes for workers to figure out what was going on and where to send fire trucks.
The first fire trucks and rescue squads were on the scene six minutes after the call was received, Ortarsen told a news conference.
He and other officials declined to comment on whether a quicker response could have saved any of the mostly immigrant victims.
But the minutes of delay felt endless to those caught in the terror of the fire and survivors have spoken angrily of what they saw as a slow and even obstructive response.
``No help.
No police.
No firemen,'' 17-year-old Zuhir Hersi, one of the disc jockeys at the bash said Friday, hours after the blaze exploded.
``Just kids helping kids.''
And once the squads arrived, the kids were then blocked from helping, they say.
``We could have saved more young people if only police hadn't stopped us,'' Mohanned Hussein was quoted as saying by the newspaper Expressen.
On Saturday, hundreds of people stood quietly outside the gutted building amid flowers and candles as they attempted to come to grips with catastrophe.
In the parking lot that a day before had been a tumult of ambulances and screams, mourners had laid a 30-meter-long (100-foot-long) pile of bouquets, candles and cards of remembrance.
The cards' inscriptions were brief _ ``I will see you in heaven,'' ``We miss you'' _ and the people who stood reading them also had few words.
``I just wanted to show my sympathy.
I think about them.
There's nothing else we can do,'' said Caroline Ericsson, who didn't know any of the fire's victims.
''It's damn difficult,'' said Connie Mesfin, who said she lost a friend in the blaze.
Lasse Gustafsson, a former Goteborg firefighter severely disfigured in an explosion, also came to the club site to try to show the victims' relatives and friends that spirit can help them pull through despair.
``I can't give them hope.
Consolation is enough,'' he said, as people nearby cast uneasy glances at his scars.
Many of those injured in the blaze may have to endure similar shocked looks the rest of their lives.

Authorities say the explosive fire quickly raised the temperature in the hall to 600 degrees (1,100 F).
The hall was packed far beyond its capacity.
Licensed to hold a maximum of 150, the hall held at least 250 and perhaps as many as 400 when the fire hit.
One of the two exit stairways was blocked by fire and there were conflicting statements from witnesses as to whether the fire came up the stairs from a lower level, or whether it spread there after breaking out in the second-floor rooms.
The worst previous fire disaster in modern Sweden was in 1978 in Boraas, when 20 people died in a hotel fire.
Goteborg, Sweden's second-largest city with about 435,000 people, is 500 kilometers (300 miles) southwest of Stockholm.

Article 7

A panicky telephone call in poor Swedish was the first word that authorities got of a fire racing through a dance hall crowded with immigrant teen-agers, delaying fire squads' response to the blaze that killed 60 and injured 162, officials said Saturday.
Per-Olof Ortarsen of Goteborg's emergency services line said the call was so hard to understand that it took three minutes for workers to figure out what was going on and where to send fire trucks.
The first fire trucks and rescue squads were on the scene six minutes after the call was received, Ortarsen said at a news conference.
He and other officials declined to comment on whether a quicker response could have saved any of the mostly immigrant victims.
The minutes of delay felt endless to those caught in the terror of the fire and survivors have spoken angrily of what they saw as a slow and even obstructive response.
''No help.
No police.
No firemen,'' 17-year-old Zuhir Hersi, one of the disc jockeys at the bash, said Friday, hours after the blaze exploded.
''Just kids helping kids.''
And once the squads arrived, the kids were blocked from helping, they say.
''We could have saved more young people if only police hadn't stopped us,'' Mohanned Hussein was quoted as saying by the newspaper Expressen.
On Saturday, hundreds of people stood quietly outside the gutted building amid flowers and candles as they attempted to come to grips with catastrophe.
In the parking lot that a day before had been a tumult of ambulances and screams, mourners had laid a 30-meter-long (100-foot-long) pile of bouquets, candles and cards of remembrance.
The cards' inscriptions were brief _ ``I will see you in heaven,'' ``We miss you'' _ and the people who stood reading them also had few words.
``I just wanted to show my sympathy.
I think about them.
There's nothing else we can do,'' said Caroline Ericsson, who didn't know any of the victims.
For Lasse Gustavsson, having the right words wasn't as important as showing his face, severely disfigured in a fire.
The former Goteborg firefighter lost his ears, his eyelids and most of his nose in a gas explosion.
By showing up, he said, he wanted to show the victims' relatives and friends that spirit can help them pull through despair.
``I can't give them hope.

Consolation is enough,'' he said, as people nearby cast uneasy glances at his scars.
Many of those injured in the blaze may have to endure similar shocked looks the rest of their lives.
Authorities say the explosive fire quickly raised the temperature in the overcrowded hall to 600 degrees (1,100 F).
The cause of the fire that broke out just before midnight Thursday remains under investigation.
Witness accounts have varied widely, with some reporting smoke coming from the cellar and others saying the fire appeared to start in the ceiling of the dance hall on the building's second floor.
The fire's quick spread has prompted speculation that it could have been set, but officials also say the explosive spread could have been because the fire had been burning undetected for some time.
What's known is that the hall was packed far beyond its capacity.
Licensed to hold a maximum of 150, the hall held at least 250 and perhaps as many as 400 when the fire hit.
The crowd was mostly teen-agers and mostly immigrants or children of immigrant parents.
They had come for a dance organized by eight party-arrangers whom police have not identified; the hall was rented by the organizers from the local Macedonian immigrant association.
Officials said the dead and injured were of 19 nationalities, including Somalis, Ethiopians, Iraqis, Iranians and Swedes, as well as people from the current and former Yugoslavia and unspecified Latin American countries.
Identifying the dead was a wrackingly slow process, forcing relatives and friends already exhausted with dread to wait for hours at hospitals.
Only 18 of the dead had been identified by midday Saturday.
``The identification is hard because they have no driver's licenses or other documents _ they were so young,'' said Kerstin Einarsson of Sahlgrenska Hospital, the largest in the city of 435,000 residents about 300 miles (500 kilometers) west of Stockholm.
The worst previous Swedish fire disaster in modern history was in 1978 in Boraas, when 20 people died in a hotel fire.

Article 8

Hundreds of teen-agers jammed into an upstairs hall planning to dance the night away, but by the time the sun rose Friday they were dead, clinging to life in hospitals or weeping in disbelief at a fire that killed 67 of them.
Police said another 173 people were injured, 20 of them severely, in the explosive fire that engulfed the plain brick two-story building just before midnight Thursday and turned a boisterous disco dance into a screaming terror in a matter of moments.
Bent metal bars on some of the hall's windows showed the panic-fueled strength that the teen-agers, many of them immigrants, exerted as they sought ways out of the hall, which with some 400 people inside was crowded to more than twice its legal capacity.
Below the second-floor windows lay stray shoes, broken glass and bloody blankets used to wrap those who may or may not have lived.
``I've been crying all day.
I haven't been able to sleep.
I'm alive, so why should I sleep when my friends are dead,'' 17-year-old Alina Turk said as she stood outside the ruined building Friday afternoon.
She said she had been at the dance and two male friends of hers died.
Zuhir Hersi, the 17-year-old disc jockey at the bash, told of his ordeal in telegraphic bursts.

``Panic.
No help.
No police.
No firemen.
Only kids helping each other,'' he said from his bed in a Goteborg hospital.
Although Hersi, like many of the youths at the dance, believes the police and firemen were slow, authorities said the first fire trucks were on the scene within five minutes of getting the alarm.
But the fire spread so fast that even an instant response would likely have been too slow.
The cause of the fire had not been determined as of Friday evening.
Fire Brigade Engineer Bo Wahlstroem said the flames' quick, raging spread could indicate arson, or that the fire had burned undetected for a time before exploding.
The fire destroyed the building, whose second floor, where the dance took place, was rented by the local Macedonian immigrant association in Sweden's second-largest city.
The association had hired out the hall for the night to eight party arrangers, police said but declined to identify the arrangers.
The crowd was mostly aged 13 to 18, witnesses said, and consisted mostly of immigrants or children of immigrant parents.
Officials said the dead and injured were of 19 nationalities, including Somalis, Ethiopians, Iraqis, Iranians and Swedes, as well as people from the current and former Yugoslavia and unspecified Latin American countries.
Identifying the dead was a wrackingly slow process, forcing relatives and friends exhausted with dread to wait for hours at hospitals.
Some were able to make the wait in rooms off-limits to journalists, but many had to wait in corridors, crying and teen-agers hurry in and out of rooms as they looked for their pals.
Only 14 bodies had been indentified by Friday evening.
``The identification is hard not only because of the burns but also because they have no driver's licenses or other documents _ they were so young,'' said Kerstin Einarsson of Sahlgrenska Hospital, the largest in the city of 435,000 some 500 kilometers (300 miles) west of Stockholm.
The blaze, the worst fire disaster in Sweden's modern history, shocked a country renowned for its smooth calmness.
King Carl XVI Gustaf, on a trip out of the country, sent a statement reassuring victims' relatives that ``all of us in Sweden feel great sympathy.''
Prime Minister Goeran Persson travelled from Stockholm to the fire site, first laying flowers outside and walking into the gutted wreck.
``The floor was full of shoes and boots, the same kind of boots my own children wear,'' he said.
On Friday evening, about 1,500 mostly young people, came to the Goteborg Cathedral to try to assuage their grief and bewilderment at a memorial service.
The youths were dressed in the same sort of hip-hop garb that the dance-goers had worn, but listened to delicate hymns instead of pounding disco.
They wept, they embraced, and some looked around nervously, apparently not knowing how to behave in a church.
``I don't go to church.
I'm a Muslim, but I don't go to prayers,'' said a young man who gave his name only as Sami.
``I'm accompanying my Christian friends.
We lost someone.
We're mourning _ that's all.''

Model Human Reference 1

At least 60 teenagers were killed and another 160 were injured in a dance hall fire in Goteborg, Sweden, Sweden's second largest city.
The fire was the worst in Sweden's modern history.
At least 400 teenagers, attending a Halloween dance, were crammed into a facility meant to hold 150.
The dance attendees were mostly immigrant children from representing 19 nationalities, including Somalia, Ethiopia, Iraq, Iran and former Yugoslavia.
The cause of the fire, which quickly engulfed the two-story brick building is unknown as investigators continue to probe the ruins.
Emergency help was delayed by about three minutes because of language difficulties.

Model Human Reference 2

A fire rapidly destroyed a Goteborg, Sweden dance hall as some 400 teenaged Halloween revelers jammed up while trying to evacuate.
At least 60 died, mostly from smoke inhalation, and about 150 were injured.
The actual capacity of the 2nd floor dance hall was 150.
Forensic experts are having success identifying the burned bodies but the question of how the fire began is still unanswered.
The blaze gutted the entire building owned by the Macedonian Association.
Mostly immigrants attended the dance.
The fire squads say they lost about six minutes while trying to respond to a call in poor Swedish.
It is doubtful a quicker response would have made a difference.

Model Human Reference 3

Sweden's deadliest fire occurred on Halloween when a dance hall in Goteborg, filled with teenagers, burst into flames.
The cause is not known, but the hall, approved for 150, contained as many as 400 and one of the two exits was blocked.
Sixty were killed and between 150 and 173 injured.
Most were immigrants or their children and represented 19 nationalities.
Many felt the rescue was late.
Rescuers said they had trouble understanding the first call, but were there in 6 minutes.
Condolences came from Swedish authorities.
A memorial service was held and a memory wall was growing.
Although identifications are difficult, 40 of the dead have been identified.

Model Human Reference 4

A fire at an overcrowded dance hall in Goteborg, Sweden killed 60 and injured 180.
Most victims were 13-18, immigrants covering 19 nationalities.
The 2-story brick building was rated for 150 but held 400.
Owned by the Macedonian Association, it had been rented out to 8 party arrangers for a Halloween dance, held on the 2nd floor.
One of two exit stairways was blocked by fire.
A panicky phone call in poor Swedish to authorities took 3 minutes to understand.
Fire trucks were on the scene 6 minutes later.
The explosive, fast-spreading fire reached 600C.
It may have burned undetected for some time.

Arson was a possibility but the cause remains undetermined.

Our Generated Summary

A fire turned a dance hall jammed with teen-age halloween revelers into a deathtrap killing 60 people and injuring about 180.
The fast-spreading fire completely gutted the building and left rescuers facing a hideous scene that local rescue service leader lennart olin likened to a gas chamber.
Police said most victims choked to death on smoke and poisonous gases 59 bodies were found at the scene and six others died later.
The macedonian association when he saw the fire.
The fire was the deadliest in modern swedish history since 1978 when 20 people died in a hotel fire.
Fire officials were alerted at 11 43 p.m 2243 gmt thursday and had a fire truck at the scene four minutes later rescue workers said.
It was a panic goteborg police spokesman bengt staaf said with youths trampling each other to get out and other youths scuffling with police to get in and attempt to help friends.
The building had just two exits one of which was blocked by fire city police technician stephen holmberg was quoted as saying by tt.
Identifying the dead was a wrackingly slow process forcing relatives and friends exhausted with dread to wait for hours at hospitals.
The crowd in the second story of the building contained mostly 13 to 18 year olds celebrating halloween and a holiday weekend.
Jamal fawz 15 told tt that he was out on the dance floor when the blaze started with about 400 people inside.
Authorities say the explosive fire quickly raised the temperature in the hall to 600 degrees 1,100 f.
The goteborg rescue services also brought city buses into service to help transport the injured.
Goteborg sweden's second-largest city with about 435,000 people is 500 kilometers 300 miles southwest of stockholm.

MEAD Generated Summary

It reminded me of the gas chambers at Auschwitz,'' local rescue service leader Lennart Olin said on national radio, describing the sight when rescuers first entered the building in Goteborg, a city of half a million people on the country's west coast.
We are still searching the building ... but so far we have found 60 dead,'' Goteborg police official Jan Edmundson said on national radio.
At a news conference Friday morning, police said most victims choked to death on smoke and poisonous gases.
The injured toll was lowered slightly to 180, but included seven people who needed treatment at emergency burn centers.
Binan Atta was walking to the Macedonian Association when he saw the fire.
He said he raced in and pulled to safety several people, including a friend.
His skin was red and bubbly,'' Atta said.
Lots of kids were just screaming,'' he added at Hammarkullen Lutheran Church, where several dozen family and friends of victims gathered.
Fire officials were alerted at 11:43 p.m. 2243 GMT Thursday and had a fire truck at the scene four minutes later, rescue workers said.
It was a panic,'' Goteborg police spokesman Bengt Staaf said, with youths trampling each other to get out, and other youths scuffling with police to get in and attempt to help friends.

The fire broke out about midnight 2300 GMT Thursday in building of the local Macedonian Association, which organized the dance.
Olin said the rescue service inspected the building in April 1997 and fulfilled all possible demands as far as emergency exits and the possibility for fast evacuation.''
A fire turned a dance hall jammed with teen-age Halloween revelers into a deathtrap, killing at least 60 people and injuring about 180 in Sweden's second-largest city.
Although an estimated 400 people, most aged 13 to 18, were at the dance on the building's second floor, the facility had approval for a maximum capacity of 150, said Hans Carlsson, the detective leading the disaster investigation.
In 1978, 20 people died in a fire at a hotel in the town of Boraas.
At a news conference Friday morning, police said most victims choked to death on smoke and poisonous gases; 59 bodies were found at the scene and the 60th victim died at a hospital, officials said.
The dance apparently was attended mostly by immigrants or children of immigrant.
Police said the list of injured included Somalis and people from current and former Yugoslavia.
The Macedonian Association, which rents space in the building, had in turn rented the facility out to another person for the dance, Carlsson said.
The building had just two exits, one of which was blocked by fire, city police technician Stephen Holmberg was quoted as saying by the Swedish news agency TT.
The fact that it spread so fast indicates that it was not a normal fire,'' he said.
The fire, at the facilities of the Macedonian Association local immigrant group, was the deadliest in modern Swedish history since 1978, when 20 people died in the town of Boraas.
The building did not have sprinklers and was not required to have them, officials said.
Police said the dead or injured represented at least 19 nationalities, including Somalis, Ethiopians, Iraqis, Iranians and Swedes, along with people from current and former Yugoslavia and unspecified Latin American countries.
A night full of expectation, happiness over extra leave from school and high spirits in anticipation of a weekend was brutally and suddenly changed into a tragedy of incomprehensible dimensions,'' the Goteborg city council said in a statement.
Jamal Fawz, 15, told TT that he was out on the dance floor when the blaze started with about 400 people inside.
It looked like it started in the ceiling, and lamps and loudspeakers fell to the floor,'' he was quoted as saying.
Others kicked out the windows and jumped out.''
Girls don't carry their identification on them, but in a bag and maybe that wasn't lying with the body,'' she said, according to TT.
Police earlier had reported 65 dead, but backed off that figure Friday evening.
''The earlier information that police gave out was wrong,'' Hans Carlsson, the lead detective in the case, told a news conference.
The fast-spreading fire that broke out just a few minutes before midnight Thursday gutted the building and left rescuers facing a hideous scene that local rescue service leader Lennart Olin likened to a gas chamber.''
The cause of the fire had not been determined as of Friday evening.
Hundreds of teen-agers jammed into an upstairs hall planning to dance the night away, but by the time the sun rose Friday they were dead, clinging to life in hospitals or weeping in disbelief at a fire that killed 67 of them.
Police said another 173 people were injured, 20 of them severely, in the explosive fire that engulfed the plain brick two-story building just before midnight Thursday and turned a boisterous disco dance into a screaming terror in a matter of moments.

Bent metal bars on some of the hall's windows showed the panic-fueled strength that the teen-agers, many of them immigrants, exerted as they sought ways out of the hall, which with some 400 people inside was crowded to more than twice its legal capacity.
Below the second-floor windows lay stray shoes, broken glass and bloody blankets used to wrap those who may or may not have lived.
I'm alive, so why should I sleep when my friends are dead,'' 17-year-old Alina Turk said as she stood outside the ruined building Friday afternoon.
She said she had been at the dance and two male friends of hers died.
Only kids helping each other,'' he said from his bed in a Goteborg hospital.
Although Hersi, like many of the youths at the dance, believes the police and firemen were slow, authorities said the first fire trucks were on the scene within five minutes of getting the alarm.
Fire Brigade Engineer Bo Wahlstroem said the flames' quick, raging spread could indicate arson, or that the fire had burned undetected for a time before exploding.
The fire destroyed the building, whose second floor, where the dance took place, was rented by the local Macedonian immigrant association in Sweden's second-largest city.
The identification is hard not only because of the burns but also because they have no driver's licenses or other documents _ they were so young,'' said Kerstin Einarsson of Sahlgrenska Hospital, the largest in the city of 435,000 some 500 kilometers 300 miles west of Stockholm.
King Carl XVI Gustaf, on a trip out of the country, sent a statement reassuring victims' relatives that all of us in Sweden feel great sympathy.''
The floor was full of shoes and boots, the same kind of boots my own children wear,'' he said.
I'm a Muslim, but I don't go to prayers,'' said a young man who gave his name only as Sami.
A panicky telephone call in poor Swedish was the first word that authorities got of a fire racing through a dance hall crowded with immigrant teen-agers, delaying fire squads' response to the blaze that killed 60 and injured 162, officials said Saturday.
Per-Olof Ortarsen of Goteborg's emergency services line said the call was so hard to understand that it took three minutes for workers to figure out what was going on and where to send fire trucks.
The first fire trucks and rescue squads were on the scene six minutes after the call was received, Ortarsen said at a news conference.
He and other officials declined to comment on whether a quicker response could have saved any of the mostly immigrant victims.
The minutes of delay felt endless to those caught in the terror of the fire and survivors have spoken angrily of what they saw as a slow and even obstructive response.
No firemen,'' 17-year-old Zuhir Hersi, one of the disc jockeys at the bash, said Friday, hours after the blaze exploded.
''We could have saved more young people if only police hadn't stopped us,'' Mohanned Hussein was quoted as saying by the newspaper Expressen.
The cards' inscriptions were brief _ I will see you in heaven,'' We miss you'' _ and the people who stood reading them also had few words.
There's nothing else we can do,'' said Caroline Ericsson, who didn't know any of the victims.
Consolation is enough,'' he said, as people nearby cast uneasy glances at his scars.
Witness accounts have varied widely, with some reporting smoke coming from the cellar and others saying the fire appeared to start in the ceiling of the dance hall on the building's second floor.
They had come for a dance organized by eight party-arrangers whom police have not identified; the hall was rented by the organizers from the local Macedonian immigrant association.

The worst previous Swedish fire disaster in modern history was in 1978 in Boraas, when 20 people died in a hotel fire.
Forensic experts examining heavily burned bodies were able Saturday to identify more of the 60 young people who died in a dance hall fire, but the catastrophe's most tormenting question was still unanswered.
How could it happen; what caused the flames that raced through a hall packed far beyond capacity, blocking one of the exits and forcing panicked teen-agers to flee down the one remaining staircase and leap out of second-story windows?
''As long as the technicians haven't established the cause of the fire, we don't know if it's arson or not,'' Goteborg chief prosecutor Ulf Noren said Saturday evening.
Earlier in the day, Noren had said it was a ''50-50'' chance that the fire was arson, prompting wide speculation that authorities had tracked down new clues.
As investigators worked to find the cause, examiners identified another 22 of the bodies, bringing the total to 40, and officials said 49 people were released from hospital.
Of the 162 people who suffered non-fatal injuries in the Thursday night fire, 76 remain hospitalized.
The first call alerting authorities to the fire was made in heavily accented Swedish and that, combined with noise and the caller's distress, delayed the fire squads' response by several minutes.
''It's damn difficult,'' said Connie Mesfin, who said she lost a friend in the blaze.
One of the two exit stairways was blocked by fire and there were conflicting statements from witnesses as to whether the fire came up the stairs from a lower level, or whether it spread there after breaking out in the second-floor rooms.