Extractive Based Email Summarization: An Unsupervised Hybrid Approach using Graph Based Sentence Ranking and K-Means Clustering Algorithm

Milind Shivolkar, Jyoti D. Pawar, S. Baskar
DCST - Goa University
Taleigao Plateau, Goa 403206
milind.shivolkar@gmail.com, jdp@unigoa.ac.in, baskar@unigoa.ac.in

Abstract

Over the years, Automatic Text Summarization is widely studied by many researchers. Here, an attempt is made to generate an automatic summary of a given text document based on an unsupervised hybrid model. The model comprises of an extractive method: a Graph-based text ranking and K-means: a clustering algorithm. Ranked sentences are obtained using the graph-theoretic ranking model here word frequency, word position, and string pattern based ranking are calculated. The K-Means algorithm generates the coherent topic clusters. Using the output of Graph-based method and K-means clusters, Sentence Importance Score(SIS) is calculated for each sentence, where top 70% ranked sentences and centralised topics of each cluster (excluding those topics which fall in the outlier zone) are used. The unsupervised hybrid approach is an attempt to inherit one of the human practice of reading and then summarizing the text in short while keeping the original insight of that text by the virtue of important sentences and keywords. The system is tested on dataset for Summarization and Keyword Extraction from Emails which on evaluation gives an average of 0.57 score on ROUGE 2.0 tool.

Keywords: Extractive Summarization, Graph Based Seantence Ranking, K-Means Clustring, Unsupervised method, Collocation Score

1. Introduction

The era of 21st century; is all about how rapidly one can grow and how efficiently one can utilize their time for the productive work and innovative development. Rapidness and efficiency of any individual depended on his/hers grasping and learning powers. The e-Mail system has become one of the important components to any individual having an on-line conversation with the other person. The e-Mail system is being drastically used in industries for having formal communication. It is also being used by the common people to have a private conversation. In industry every individual have to deal with their clients,team lead,boss and many other things via e-Mails. With such a pressure cooker situation Automatic Summarization Tool will always be handy giving a gist of every mail from the inbox.

The natural language always consists of multiple information; among them 1: surface information: where one can capture just by reading the text and 2: hidden information: where one need to understand the contextual information hidden in the given text along with the surface information. In this paper, we are dealing with the surface information, where the attempt is made to use the human phycology of picking up the key sentences and keywords to generate the summary.

Here we have chosen an e-Mail domain considering it as one of a small subdomain but an important aspect of the on-line text for every individual who are connected to the e-Mail system. As e-Mail forms the major means of formal communication across all forms of industry, a lot of text is being generated on every individual e-Mail portal.

2. Related Work

A need of automatic text summarization was felt in 1958 by (Luhn, 1958) where the attempt was made to summarize the technical document that describes the research at IBM in the 1950s. The frequency of each word was counted; this count was a measuring factor of finding the word usefulness in the article. Each word was initially stemmed and later they were indexed in the decreasing frequency. The index provided the significance level of the words. At a sentence level, a significance factor was derived that reflects the number of occurrences of significant words within a sentence and the linear distance between them due to the intervention of non-significant words. Ranking of all sentences was done on the basis of a significant factor and top ranked sentences were selected to be the part of an auto-summarized abstract.

Later (Baxendale, 1958) in his work concluded that positional feature plays an important role in selecting the topic-oriented sentences. The author examined 200 paragraphs where the findings were such that in 85% position appeared as the topic sentence and in 7% of the paragraph the topic sentence had appeared at last position. Thus, one of the better ways is to select the topic sentences from these two positions and since then, these positional features have been used in many complex machine learning based systems.

(Edmundson, 1969) work had added two new features to Automatic Text Summarization in addition to existing previous work features. Two new features used were: the presence of cue words (presence of words like significant, or hardly), and the skeleton of the document (whether the sentence is a title or heading). The development of a typical structure for an extractive summarization experiment was the primary contribution of the author.

With the emergence of Machine Learning (ML) techniques along with Natural Language Processing (NLP) in 1990, a series of seminal publications appeared that employed statistical techniques to produce document extracts. Most of the system initially relied on Naive-Bayes methods which were independent of features; others have focused on the choice of appropriate features and on learning algorithms

that make no independence assumptions. Hidden Markov models (Conroy and O'leary, 2001) and log-linear models (Osborne, 2002) where the significant approaches which improved the extractive summarization. In contrast, use of neural networks and third party features (like common words in search engine queries) was made to improve purely extractive single document summarization. Graph-based ranking methods were introduced for sentence extraction; which primarily ranks the sentences. In this approach, an effort was made to capture the silent features of the text. Page Graph-based ranking algorithms, such as Kleinberg's HITS algorithm (Kleinberg, 1999) or Google PageRank (Page et al., 1999), have been traditionally and successfully used in citation analysis, social networks, and the analysis of the link structure of the World Wide Web. In short, a graph-based ranking algorithm is a way of deciding on the importance of a vertex within a graph, by taking into account global information recursively computed from the entire graph, rather than relying only on local vertex-specific information. On similar grounds, the Graph based ranking called TextRank (Mihalcea and Rada, 2004), can be applied for lexical or semantic graph extracts.
(Radev et al., 2000) in his work claimed that centroids play central role in summarization. Later (Radev et al., 2004) developed a centroid-based approach called MEAD system. Recently in (Ingole et al., 2012) used the Expectation-Maximization (EM) algorithm to find out sentence similarity along with Natural Language Processing (NLP) techniques at the initial stages. The findings of this paper were that their technique was better than the previous state of art approaches in terms of time and space consumption.

3. Proposed System

The proposed system is divided in two parts; one comprising of a Graph Based Sentence Ranker and other consisting of K-Means clustering algorithm producing topic clusters The proposed system works on a general methodology of how a human being would try to produce a summary of a given text. The methodology is divided in two parts:

- Finding important sentences. (As humans remembers the important sentences)
- Finding topics based on its information and importance levels.(As humans remembers the key words)

3.1. Graph Based Sentence Ranker

The method focuses on obtaining the sentence rank based on the word frequency, word position, and string pattern. The entire text is represented as a weighted undirected complete graph G (V,E); where vertex represents sentence and edges represents the similarity between each sentence. This method is further divided into two parts as follows:

3.1.1. Sentence Weight

An affinity weight is calculated to find relativeness of each word in the text document based on the word frequency score. This ensures the importance of each word in the text document. An affinity weight (AW) of each word is stated as a ratio of the occurrence of each word (w_i) in given document D with the number of words ($N(w)$) in the given document D; its mathematical formulation is given as

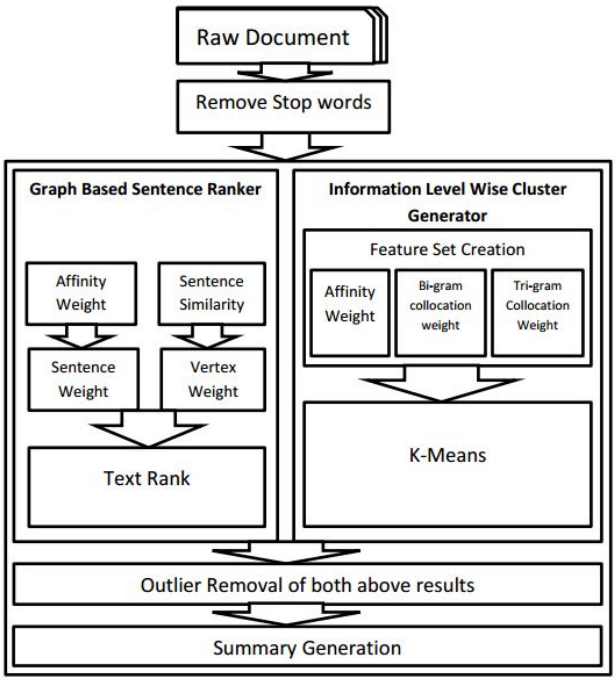


Figure 1: A block diagram of Extractive Email Summarization using An Unsupervised Hybrid Approach of Graph Based Sentence Ranking and K-Means Clustering Algorithm

$$AW(w_i) = \frac{n(w_i)}{N(w)}$$

Sentence weight is calculated using above affinity weights obtained for each word(wi). Each sentence comprises of n number of words and each word knows its own affinity weight; summing over the n affinity weight in the sentence S_i will produce the sentence score for sentence S_i . The ratio of sentence scores with the number of words in sentence S_i will give the sentence weight (SW). The mathematical formulation of sentence weight (SW) is given as

$$SW(S_j) = \frac{\sum_{w_i \in S_j} AW(w_i)}{n(S_j)}$$

3.1.2. Vertex Weight

Levenshtein Similarity Weight (LSW) is calculated to find the similarity distance between two sentences Levenshtein Similarity Weight (LSW) is calculated as the ratio of difference between the maximum length of two sentences and Levenshtein Distance(LD) with the maximum length of the two sentences and it is formulated as

$$LSW(S_i, S_j) = \frac{maxLen(S_i, S_j) - LD(S_i, S_j)}{maxLen(S_i, S_j)}$$

Given a sentence S_i; its Levenshtein Similarity Weight(LSW) is calculated with all remaining $S_(N-1)$ sentences from the document D. As discussed, set of edges(E) represent the similarity weights between sentences and set of vertices(V) represent sentences. The

vertex weight of vertex V_i is computed as an average of all the edges associated with vertex V_i ; mathematically vertex weight is written as .

$$VW(S_i) = \frac{\sum_{\forall S_i \neq S_j \in D} LSW(S_i, S_j)}{n(S)}$$

Sentence rank(SR) is obtained by averaging the results of Sentence Weight (SW) and Vertex Weight (VW) of each sentence S_i as shown in the equation below; where, $SW(S_i)$ gives the information of sentence importance in the given document based on affinity weight and $VW(S_i)$ provides the information of largest common subsequence using LSW.

$$SR(S_i) = \frac{SW(S_i) + VW(S_i)}{2}$$

In this method, the proposed system gains the knowledge of an important sentences from a given document.

3.2. Information Level Wise Cluster Generation

The second phase of the proposed system is to find the important topics from the input text document D. The K-Means clustering algorithm have been used in order to find the topic clusters of the given text document D. On a successful run of this step we obtain K clusters where each cluster signifies its level of importance in the text document D. Each text document consists of an information which spreads across the document. A major part of a document revolves around the core idea and an important information; but there are other contextual and related information present across the document making a core idea to be sensible and meaningful. The level of information of every topic is defined by its collocation scores. Collocation score of words are used as an input feature-set to K-Means algorithm, each cluster form contains a certain level of collocation information. For instance, if size of $K = 3$ in K-Means algorithm then the obtained clusters will have three levels of information :

- High : Cluster having highest collocation words together,
- Medium : Cluster having medium collocation words together,
- Low : Cluster having low collocation words together.

The K-Means clustering algorithm takes a numerical data as an input to produce output. Hence, the given text data is converted in numerical form having three features; an affinity score, an expected bigram collocation score and an expected trigram score.

3.2.1. Dataset

The dataset consists of three features for a given word (w_i) in document D:

Affinity Score: This score gives the information of word importance in given document D.

Expected Bigram Collocation Score EBCS: Collocations are expressions of multiple words which commonly co-occur. Similarly, bigram collections are an expression consisting of two words which commonly co-occur. The dataset intends to find the importance of a word (w_i) in the given document D; therefore, we compute the average bigram collocation score which gives an expected score of the word (w_i) with respect to document D.

$$EBCS(w_i) = \frac{\sum_j BigramCollocationScore(w_i, w_{i,j})}{Count(BigramCollocation(w_i, w_{i,j}))}$$

where w_i coreesponds target word, $w_{i,j}$ corresponds to word co-occurring with target word and j range from 0 to m; where m represents the number of co-occurring words with respect to target word w_i.

An expected bigram collocation can be looked upon as the following observation; let us consider a classroom where there are many objects having different properties and its usage. For instance, let's consider an object; a Ram's water bottle. A water bottle can contain water, juice or other liquid; it can also be empty or full; it can be used by Ram or by his friends and many such things can co-occur with Ram's water bottle. In order to find the expected importance of Ram's, water bottle in a classroom one can average all such co-occurring scores. Similarly, we treat classroom as a document, Ram's water bottle as a word (w_i) and all other co-occurring relation with Ram's water bottle as co-occurring words $w_{i,j}$ with the given word(w_i); hence the Expected Bigram Collocation Score.

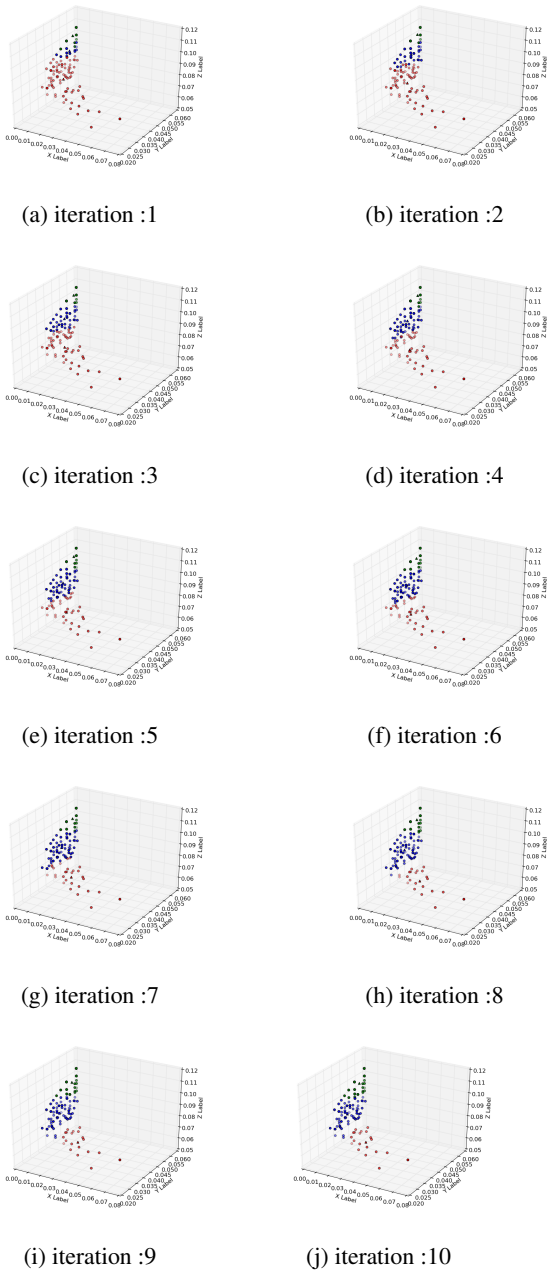
Expected Trigram Collocation Score ETCS : This score are generated on similar basis as Expected Bigram Collocation Score. Only difference of this feature is that a word w_i co-occurs with a pair of words $w_{i,k}$.

$$ETCS(w_i) = \frac{\sum_k TrigramCollocationScore(w_i, w_{i,k})}{Count(TrigramCollocation(w_{i,k}))}$$

Where $w_{i,k}$ is the co-orccuring pair of words and k represents the range $i.e.$ (0 to m) of pair of words that co-occur with word w_i

3.2.2. Averaged Elbow Method:

In K-means algorithm, the document D is treated as beg of words containing n words. All words are considered to be likely distributed. Random k value is chosen to initiate the algorithm. The choice of an approximate size of K is a decisive factor for the successful run of K-Means algorithm. Elbow method is one of the known techniques used to find the size of K. Plotting the percentage variance explained by clusters against the number of clusters will produce an elbow shaped graph on the $x - y$ scale. At the start when the K value is less the clusters add much information but as it increases at some point the marginal gain drops, giving an angle in the graph. This point forming an angle on the $x - y$ graph is called as elbow and its corresponding value on $x - axis$ is chosen as K value for K-Means algorithm. In order to find the exact estimation of K value; the elbow method is iterated over for N times and average elbow plot is obtained which gives the correct estimation of K.



(a) iteration :1 (b) iteration :2

(c) iteration :3 (d) iteration :4

(e) iteration :5 (f) iteration :6

(g) iteration :7 (h) iteration :8

(i) iteration :9 (j) iteration :10

Figure 2: Cluster plots from $(a) - (j)$ shows the cluster formation for $k = 3$ of a random document D. At every iteration from $(a) - (j)$ the K-Means algorithm is clustering the words groups having simillar collocation score. At 10^{th} iteration we can see the 3 clusters formed are based on thier feature-set score that includes affinity weights and collocation scores. The cluster formed at 10^{th} iteration visually shows the 3 level of information that is being captured in 3 clusters.

3.2.3. Clustering Output

The K-Means algorithm is used to partition the n observation of the dataset into k clusters. The second part of the proposed system is to find important topics. The dataset created contains the word importance scores for each word w_i in the document D. On successful run; the K-Means clustering algorithm gives k clusters. Each cluster formed is based on the level of importance (a collocation factor) the word w_i have in document D. "Level of importance" is a vital feature for any human being; for instance, many in India will remember "Sachin Tendulkar" as a cricket player due to his immense contribution to Indian cricket but very few will remember "Robin Uthappa". Considering above instance and also knowing the craze of cricket in India if we ask any Indian about "Indian Cricket" then, more often than not the co-occurring reply will be "Sachin Tendulkar". In this case, for keyword "Indian Cricket", there is a high probability that a keyword "Sachin Tendulkar" will co-occur more frequently than the keyword "Robin Uthappa". In above instance, we see that "Sachin Tendulkar" and "Robin Uthappa" co-occur with "Indian Cricket" explaining how "Level of Importance" influences the choice of a human being. On similar basis if we consider a keyword "World Cricket" the impact of "Sachin Tendulkar" will be much higher than that of "Robin Uthappa" as the gap of co-occurrence between '"world Cricket" - "Sachin Tendulkar"' and '"world Cricket" - "Robin Uthappa"' will increase exponentially. The gap increases exponentially since there are many legendary cricketers across the world and as Sachin himself is one of the legends and Robin is not.

From above discussion, we can say, that "Sachin Tendulkar" forms a global keyword and "Robin Uthappa" on a local keyword. The dataset discussed above makes sure that it captures all the co-occurrence features by using bigram, trigram Collocation features from the given document. Obtained K size from elbow method defines the number of clusters. Each cluster formed gives us a group of keywords that falls in one "Level of Importance".For instance, if K=3 then the output of K-Means will give 3 clusters where each cluster will be having keywords belonging to the same "Level of Importance" in the given document D. Now we know that a document D is consists of the mixture of words, knowing this we also say that a document D will also have words which may fall in global and local keywords category. Such keywords help us to define the "level of Importance" of the word in a given document. The dataset designed comprises of co-occurrence scores which will help K-Means algorithm to form the clusters based on "Level of Importance" of a word in given document D. The size of K defines the number of clusters and each cluster will comprise all words belonging to same "Level of Importance".

The purpose of K-Means algorithm to obtain the topics clusters based on the level of word importantance was achieved.

3.3. Outlier Removal

Graph-based Sentence Ranker outputs the ranked sentences. Top 60% of ranked sentences are considered for further processing and remaining 40% are treated as outliers (less important). Similarly, K-Means output gives the topic cluster based on the word level importance. Computing the distance between the centroid and furthest point in that cluster gives the maximum radial distance of the cluster. The cluster points that falls within the 70% of the maximal radial distance (MRD) are used for further processing and remaining are treated as outliers. In order to summarise any document, we use most important keywords, phrases

or sentences and hence discarding the less important data as outliers are oblivious.

3.4. Summary Generation

Summary Generation consists of two steps as follows:

3.4.1. Text Ranking

The clustering algorithm outputs the clusters of word level importance. These clusters are used to estimate the sentence importance score (SIS). The sentence importance score is calculated as below:

$$SIS(S_i) = SR(S_i) * CW(S_i)$$

where Cluster Weigth(CW) of sentence S_i is writen as below:

$$CW(S_i) = \frac{\sum_i MRD(w_i)}{C}$$

where C = number of cluster S_i belongs to.
Higher the value of CW more important is the sentence.

3.4.2. Summary Generation

In Text Ranking step we obtain the final ranking score of each sentence. In this step, a sorting is performed on the sentences based on their score. We aim to produce an extractive summary of 1 to 8 sentences for every document. Depending upon the size of document predefined rule has been defined. For document having less the 25 sentences, we decided to keep top 40% of the sentences while for document larger than 25 sentences a threshold of top 5-8 or 5% sentences has been kept (whichever condotion yeild less sentences will be chossen as output). The sentence order has been taken care by the sentence index which can be used to resort the top sentences.

4. Analysis and Evaluation

For experimental testing, two categories of email are being used. The first category is a formal co-operate email conversation consisting of single mail and second category comprises of informal personal email conversation over a single mail. Based on the obtained result we analysed that informal conversation rarely happens on a specific set of topics on other hand the formal conversation contains its own limits resulting in focus concersation.

Document Category	ROUGE-1 Score	ROUGE-2 Score
Formal Emails	0.59	0.57
Informal Emails	0.49	0.35

Table 1: ROUGE score

The experimental result of the proposed system is evaluated using ROUGE 2.0. The testing of the system is done using a "Dataset for Summarization and Keyword Extraction from Emails (Loza et al., 2014)". As discussed above, two sets of email conversation were taken. The resulted system summary for formal conversation gives the ROUGE-2 score of up to 0.86 and on an average, the system score for formal mail conversation set is 0.57. On the other hand, system heavily suffered on informal mail conversation. The highest score recorded is up to 0.53 and the average score lowering to 0.35.

5. Conclusions and Future Scopes

K-means plays an important role in identifying the cluster groups of words in document D. Each cluster shows its own dominance in document D at its own level. The dominance information is captured to obtain the extractive summary from ranked sentences obtained by Graph-Based Sentence Ranker.

Pre-processing steps like stemming, POS tagging and dependency parsing were not performed on the raw data. For future, we need to incorporate these steps and find its impact on the end result.

Another module of sentiment score generator could be added to the system resulting into subjective based summarization. A paraphrase module can be used on extracted summary to generate the abstractive summary.

The proposed system currently is working for English language and can easily be extended to various other languages. Especially for Indian languages where most of them lack the pre-processing tools needed for automatic text summarization. This tool can be easily used for any given language as a text summarizer with some needed minor changes to it.

6. Bibliographical References

Baxendale. (1958). Machine-made index for technical literature - an experiment. In *IBM Journal of Research Development*, pages 354–361.

Conroy and O'leary. (2001). Text summarization via hidden markov models. In *In Proceedings of SIGIR '01*, pages 406–407, New York, NY, USA.

Edmundson. (1969). New methods in automatic extracting. In *Journal of the ACM*, pages 264–285.

Ingole, Bewoor, and Patil. (2012). Text summarization using expectation maximization clustering algorithm. Citeseer.

Kleinberg. (1999). Authoritative sources in a hyperlinked environment. In *J. ACM*, pages 604–632.

Loza, Lahiri, Mihalcea, and Lai. (2014). Building a dataset for summarization and keyword extraction from emails. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, Reykjavik, Iceland, May.

Luhn. (1958). The automatic creation of literature abstracts. In *IBM Journal of Research Development*, pages 159–165.

Mihalcea and Rada. (2004). Graph-based ranking algorithms for sentence extraction, applied to text summarization. In *Proceedings of the ACL 2004 on Interactive poster and demonstration session*, page 20. Association for Computational Linguistics.

Osborne. (2002). Using maximum entropy for sentence extraction. In *In Proceedings of the ACL'02 Workshop on Automatic Summarization*, pages 1–8, Morristown, NJ, USA.

Page, Lawrence, Brin, Sergey, Motwani, Rajeev, Winograd, and Terry. (1999). The pagerank citation ranking: bringing order to the web. Stanford InfoLab.

Radev, Jing, and Budzikowska. (2000). 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*, pages 21–30. Association for Computational Linguistics.

Radev, Dragomir, Allison, Timothy, Blair-Goldensohn, Sasha, Blitzer, John, Celebi, Arda, Dimitrov, Stanko, Drabek, Elliott, Hakim, Ali, Lam, Wai, Liu, Danyu, et al. (2004). Mead-a platform for multidocument multilingual text summarization.