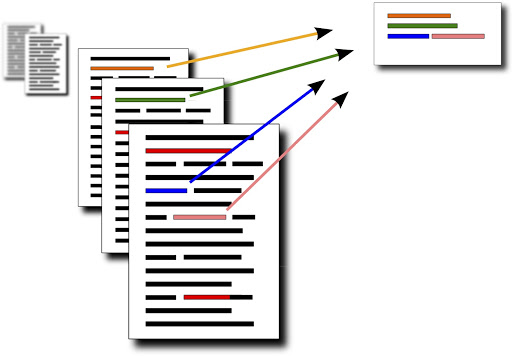
**[ AI PROJECT ]**



**Generic Text and Youtube Videos Summarization**

1. **Abstract:**

At present, usually a keyword-based search Internet returns a large number of results. This makes the user get exhausted therefore there is a higher requirement for advancement that can help the user to sieve large information, and to quickly find the most relevant documents. This research article will include studies about Latent semantic analysis (LSA) which is a technique in natural language processing (NLP). LSA assumes that words that are close in meaning will occur in similar pieces of text. LSA can use a term-document matrix which describes the occurrences of terms in documents. Latent Semantic Indexing (LSI) helps overcome synonymy by increasing recall, one of the most problematic constraints of Boolean keyword queries and vector space models. Synonymy is often the cause of mismatches in the vocabulary used by the authors of documents and the users of information retrieval systems. We have two methods i.e. Latent Semantic Analysis and Relevance Measure for finding the summary of text. The Latent Semantic Analysis uses Singular value decomposition on matrix A to get three matrices U, Σ, VT. Each value of matrix A represents whether this ith word is present in jth sentence. The U matrix representing terms to concepts relation. Each value in matrix U represents how strongly term ith is related to jth concept. The matrix Σ is a diagonal matrix representing singular values sorted in decreasing order. Each diagonal value of matrix Σ represents strength of each concept. The matrix VT is called right singular vector matrix. It represents ith concept and jth sentence. Each value in matrix VT represents some expression value or we can say some connection between concept and a sentence. From matrix VT, we will pick those sentences that are highly connected to 1st concept and that would give us the summary of the text.

Another method that is used is Relevance Measure. In this we summarize the text using the relevancy of a particular sentence in the given text. The first challenge is to break the given text into separate sentence and the tokenization of the sentences is completed the next challenge is to remove the stop words from the sentences without affecting the original sentence then we need to build the sentence vector for every sentence and store it in the suitable data structure then we need to take the dot product of the sentence vector Si with document vector D excluding the sentence i where i is the particular sentence of whom the dot product with the document is taken.

                  Score of Sentence (I )= D.Si

Then we after storing the score of each sentence we need rank them in order of the score in descending and return the required no of sentences to the user with the highest score.

We will be fetching the subtitles of a YouTube video and then we will apply the two methods to get the summary. Every text, article, paragraph or document contains sentences that represent the main topic and some other sentences that represent supporting topics to our main topic. By generating the summary what we get is list of sentences that best represent the main topic in decreasing order.

1. **Introduction:**

The explosive growth of the world-wide web has dramatically increased the speed and the scale of information dissemination. With a vast sea of accessible text documents on the Internet, conventional AI and IR technologies have become more and more insufficient finding relevant information effectively. Nowadays, it is quite common that a keyword-based search on the Internet returns hundreds, or even thousands of hits, by which the user is often overwhelmed. Therefore, there is an increasing need for new technologies that can help the user to sift through vast volumes of information, and to quickly identify the most relevant documents.

With a large volume of text documents, presenting the user with a summary of each document greatly facilitates the task of finding the desired documents. Text search and summarization are the two essential technologies that complement each other. Text search engines return a set of documents that seem to be relevant to the user's query, and text summarizers produce document summaries that enable quick examinations through the returned documents. Text search engines serve as information alters that sift out an initial set of relevant documents, while text summarizers serve as information spotters that help users to spot the final set of desired documents.

Text summaries can be either query-relevant summaries or generic summaries. A query-relevant summary presents the contents of the document that are closely related to the initial search query. Creating a query-relevant summary is essentially a process of retrieving the query relevant sentences/passages from the document.Therefore, query relevant summarization is often achieved by extending conventional IR and AI technologies, and to date, a large number of text summarizers in the literature fall into this category. On the other hand, a generic summary provides an overall sense of the document's contents. A good generic summary should contain the main topics of the document while keeping redundancy to a minimum. As no query nor topic will be provided to the summarization process, it is challenging to develop a high quality generic summarization method, and is even more challenging to objectively evaluate the method. In this project, we build an application to propose two generic text summarization methods that create text summaries by ranking and extracting sentences from the original documents. The first method uses standard IR methods to measure sentence relevancies, while the second method uses the latent semantic analysis technique to identify semantically important sentences, for summary creations. Both methods strive to select sentences that are highly ranked and different from each other. This is an attempt to create a summary with a wider coverage of the document's main content and less redundancy.

1. **Related work:**

Text summarization has been researched in recent years actively. A majority of the research studies in the literature have been focused on creating query-relevant text summaries. M. Sanderson proposed a query-relevant summarizer that divides the document into equally sized overlapping passages, and uses the INQUERY text search engine to obtain the passage that best matches the user's query. This best passage is then used as a summary of the document [1].

A query expansion technique called Local Context Analysis (LCA, which is also from INQUERY) is used before the best passage retrieval. Given a topic and a document col- lection, the LCA retrieves top-ranked documents from the collection, examines the context surrounding the topic terms in each retrieved document, and then selects and adds the words/phrases that are frequent in this context to the query.

B. Baldwin and T.S. Morton developed a summarizer that selects sentences from the document until all the phrases in the query are covered. A sentence in the document is considered to cover a phrase in the query if they co-refer to the same individual, organization, event, etc [2].

R. Barzilay and M. Elhadad developed a method that creates text summaries by finding lexical chains from the document [3]. The Cornell/Sabir system uses the document ranking and passage retrieval capabilities of the SMART text search engine to effectively identify relevant passages in a document [4].

The text summarizer from CGI/CMU uses a technique called Maximal Marginal Relevance (MMR) which measures the relevance of each sentence in the document to the user provided query, as well as to the sentences that have been selected and added into the summary [5].

The text summary is created by selecting the sentences that are highly relevant to the user's query, but are different from each other. The SUMMARIST text summarizer from the University of Southern California strives to create text summaries based on the equation: summarization=topic identification+ interpretation + generation. The identification stage alters the input document to determine the most important, central topics. The interpretation stage clusters words and abstracts them into some encompassing concepts. Finally, the generation stage generates summaries either by outputting some portions of the input, or by creating new sentences based on the interpretation of the document concepts [6]. However, this generation function was not realized in the pa- per. The Knowledge Management (KM) system from SRA International, Inc. extracts summarization features using morphological analysis, name tagging and co-reference resolution. They used a machine learning technique to deter- mine the optimal combination of these features in combination with statistical information from the corpus to identify the best sentences to include in a summary [7].

1. **Features:**

* First we fetched the data if the data isn’t available using youtube link provided.
* Then we did preprocessing i.e we first remove the stopwords then we remove characters that do not belong to English alphabets.
* Lastly we did lowercasing of all tokens.
* Every text, article, paragraph or document contains sentences that represent the main topic and some other sentences that represent supporting topics to our main topic.
* By generating the summary what we get is list of sentences that best represent the main topic in decreasing order.
* Our features are basically words in a document.
* There wasn’t any training required as this comes under unsupervised learning.

1. **Methods:**

**5.1 Summarization by Relevance Measure**

After the given document is decomposed into individual sentences, we compute the relevance score of each sentence with the whole document. We then select the sentence k that has the highest relevance score, and add it to the summary. Once the sentence k has been added to the summary, it is eliminated from the candidate sentence set, and all the terms contained in k are eliminated from the original document. For the remaining sentences, we repeat the steps of relevance measure, sentence selection, and term elimination until the number of selected sentences has reached the predefined value. The operation flow is as follows:

1. Decompose the document into individual sentences, and use these sentences to form the candidate sentence set S.
2. Create the weighted term-frequency vector Ai for each sentence i belongs to S, and the weighted term-frequency vector D for the whole document.
3. For each sentence i belongs to S, Compute the relevance score between Ai and D, which is the inner product between Ai and D.
4. Select sentence k that has the highest relevance score, and add it to the summary.
5. Delete k from S, and eliminate all the terms contained in k from the document.
6. If the number of sentences in the summary reaches the predefined value, terminate the operation; otherwise, go to Step 3.

In Step 4 of the above operations, sentence k that has the highest relevance score with the document is the one that best represents the major content of the document. Selecting sentences based on their relevance scores ensures that the summary covers the major topics of the document. On the other hand, eliminating all the terms contained in k from the document in Step 5 ensures that the subsequent sentence selection will pick the sentences with a minimum overlap with k. This leads to the creation of a summary that contains little redundancy.

**5.2 Summarization By Latent Semantic Analysis**

We here applied the singular value decomposition (SVD) to generic text summarization. The process starts with the creation of a terms by sentences matrix A = [A1 A2 … An] with each column vector Ai representing the weighted term-frequency vector of sentence i in the document under consideration. If there are a total of m terms and n sentences in the document, then we will have an m x n matrix A for the document. Since every word does not normally appear in each sentence, the matrix A is usually sparse.

Given an m x n matrix A, where without loss of generality m>=n, the SVD of A is defined as:

A = U∑V^T

where U = [uij ] is an m x n column-orthonormal matrix whose columns are called left singular vectors;∑ = diag( o1, o2, … , n) is an n x n diagonal matrix whose diagonal elements are non-negative singular values sorted in descending order, and V = [vij ] is an n x n orthonormal matrix whose columns are called right singular vectors. If rank(A)=r, then ∑ satisfies

o1 ≥ o2 . . . ≥ or > or+1 = . . . = on = 0: (3)

The interpretation of applying the SVD to the terms by sentences matrix A can be made from two different view- points. From transformation point of view, the SVD derives a mapping between the m-dimensional space spanned by the weighted term-frequency vectors and the r-dimensional singular vector space with all of its axes linearly-independent. This mapping projects each column vector i in matrix A,

which represents the weighted term-frequency vector of sentence i, to column vector i = [vi1 vi2 . . . vir ]^T of matrix V^T , and maps each row vector j in matrix A, which tells the occurrence count of the term j in each of the documents, to row vector αj = [uj1 uj2 . . . ujr ] of matrix U. Here each element vix of i, ujy of αj is called the index with the xth, yth singular vectors, respectively.

From semantic point of view, the SVD derives the latent semantic structure from the document represented by matrix A. This operation reflects a breakdown of the original document into r linearly-independent base vectors or concepts. Each term and sentence from the document is jointly indexed by these base vectors/concepts. A unique SVD feature which is lacking in conventional IR technologies is that the SVD is capable of capturing and modeling interrelationships among terms so that it can semantically cluster terms and sentences. Consider the words doctor, physician, hospital, medicine, and nurse. The words doctor and physician are synonyms, and hospital, medicine, nurse are the closely related concepts. The two synonyms doctor and physician generally appear in similar contexts that share many related words such as hospital, medicine, nurse, etc. Because of these similar patterns of word combinations, the words doctor and physician will be mapped near to each other in the r-dimensional singular vector space. Furthermore, if a word combination pattern is salient and recurring in a document, this pattern will be captured and represented by one of the singular vectors. The magnitude of the corresponding singular value indicates the importance degree of this pattern within the document. Any sentences containing this word combination pattern will be projected along this singular vector, and the sentence that best represents this pattern will have the largest index value with this vector. As each particular word combination pat- tern describes a certain topic/concept in the document, the facts described above naturally lead to the hypothesis that each singular vector represents a salient topic/concept of the document, and the magnitude of its corresponding singular value represents the degree of importance of the salient topic/concept.

Based on the above discussion, we propose the following SVD-based document summarization method.

1. Decompose the document D into individual sentences, and use these sentences to form the candidate sentence set S, and set k = 1.
2. Construct the terms by sentences matrix A for the document D.
3. Perform the SVD on A to obtain the singular value matrix , and the right singular vector matrix VT . In the singular vector space, each sentence i is represented by the column vector αi = [vi1 vi2 . . . vir ]^ of VT .
4. Select the k'th right singular vector from matrix VT .
5. Select the sentence which has the largest index value with the k'th right singular vector, and include it in the summary.
6. If k reaches the predefined number, terminate the operation; otherwise, increment k by one, and go to Step 4.

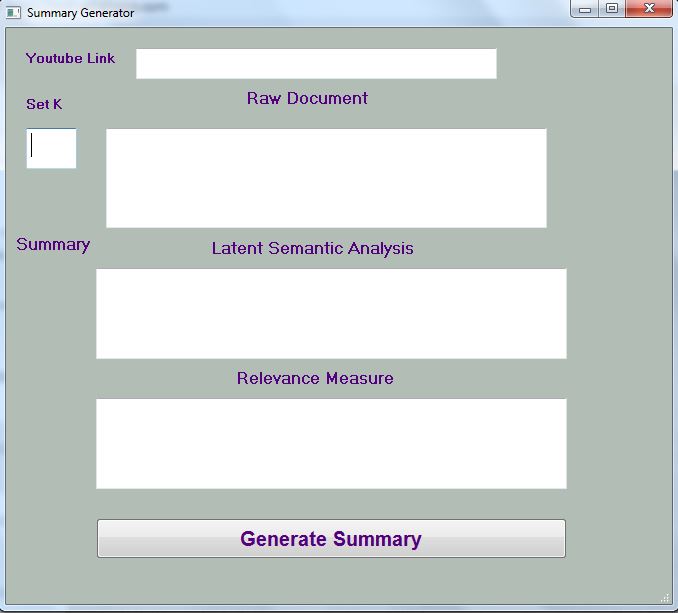
In Step 5 of the above operation, finding the sentence that has the largest index value with the k'th right singular vector is equivalent to finding the column vector i whose k'th element vik is the largest. By the hypothesis, this operation is equivalent to finding the best sentence describing the salient concept/topic represented by the k'th singular vector. Since the singular vectors are sorted in descending order of their corresponding singular values, the k'th singular vector rep- resents the k'th important concept/topic. Because all the singular vectors are independent of each other, the sentences selected by this method contain the minimum redundancy.

1. **Our Application:**

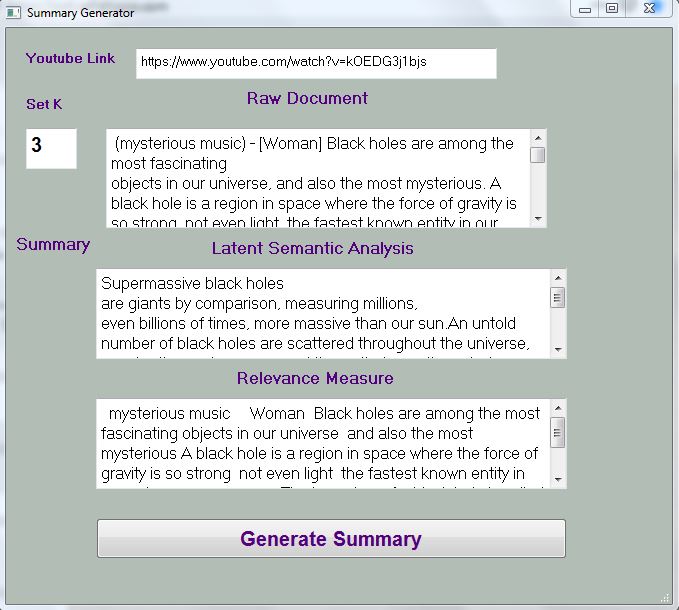
In our application we are using both the methods (Latent semantic Analysis & Relevance Measure) to summarize the texts and display it in a proper format.

We can either enter the youtube link as an input or a raw text document to summarize it.

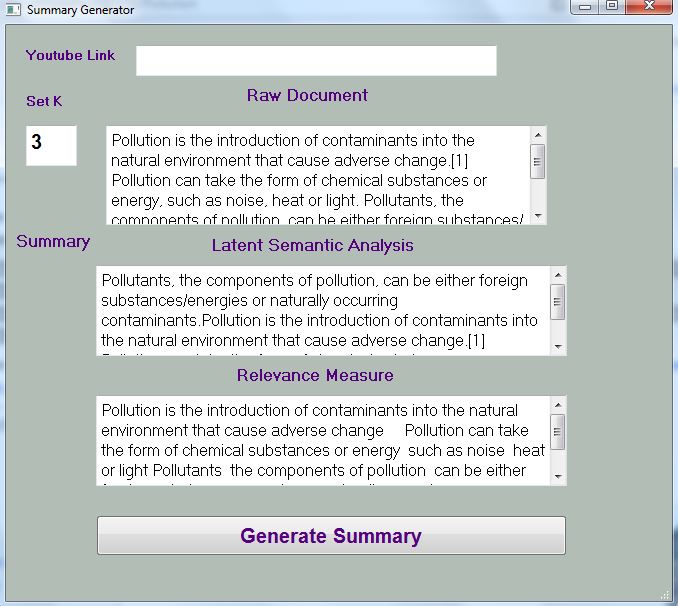
**GUI:**

****

**Using Youtube Link:**

****

**Using a raw text document:**

****

1. **Experiments/Results:**

**7.1 Performance Evaluations**

We used the recall (R), precision (P), along with F to measure the performances of the two summarization methods. Let Sman , Ssum be the set of sentences selected by the human evaluator(s), and the summarizer, respectively. The standard definitions of R, P, and F are defined as follows:

R =

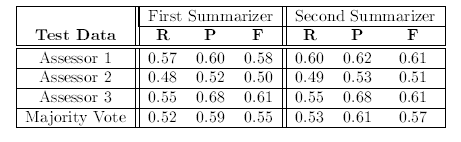
P =

F =

For our evaluations, we set the length of the machine generated text summaries to the length of the corresponding manual summaries. When the evaluation is performed using each individual manual summarization result, both |Sman | and |Ssum| are equal to five. When the evaluation is performed using the combined result determined by a majority vote, |Sman| becomes variable, and |Ssum | is set to the value of |Sman|.

The evaluation results are shown in Table. These results are generated using the BNN weighting scheme

**Table of Evaluation Results**

**

As evidenced by the results, despite the very different approaches taken by the two summarizers, their performance measures are quite compatible. This fact suggests that the two approaches interpret each other. The first summarizer the one using the relevance measure takes the sentence that has the highest relevance score with the document as the most important sentence, while the second summarizer (the one based on the latent semantic analysis) identifies the most important sentence as the one that has the largest index value with the most important singular vector. On the other hand, the

first summarizer eliminates redundancies by removing all the terms contained in the selected sentences from the original document, while the second summarizer suppresses redundancies by using the k’th singular vector for the k h round of sentence selection. The first method is straightforward, and it is relatively easy for us to give it a semantic interpretation. As for the second method, there has been a long his- tory of arguments about what essentially each of the singular vectors represents when a collection of text (which could be sentences, paragraphs, documents, etc) are projected into the singular vector space. Surprisingly, the two different methods bring to us very similar summarization outputs. This mutual resemblance enhances our belief that each important singular vector does capture a major topic/concept of a document, and two different singular vectors do capture two semantically independent topics/concepts that have the minimum overlap.

**7.2 Weighting Schemes**

In our performance evaluations, we studied the influence of different weighting schemes on the summarization performances as well. As shown by Eq.(1), given a term i, its weighting scheme is de ned by two parts: local weighting L(i) and global weighting G(i). Local weighting L(i) has the following four possible alternatives:

1. No weight: L(i) = tf(i) where tf(i) is the number of times term if occurs in the sentence.
2. Binary weight: L(i) = 1 if term i appears at least once in the sentence; otherwise, L(i) = 0.
3. Augmented weight: L(i) = 0:5+ 0:5 (tf(i)/tf (max)) where tf(max) is the frequency of the most frequently occurring term in the sentence.
4. Logarithm weight: L(i) = log(1 + tf(i)).

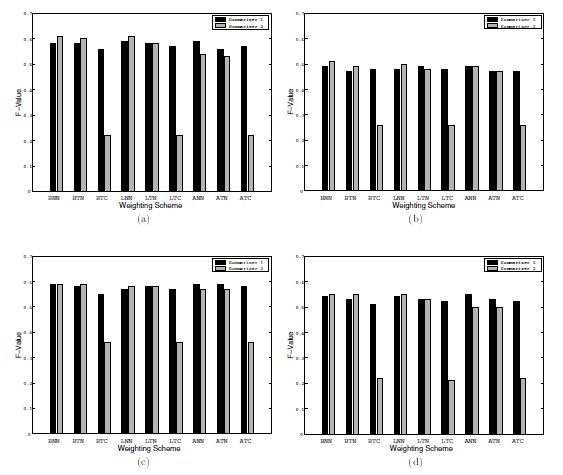
Possible global weighting G(i) can be:

* No weight: G(i) = 1 for any term i.
* Inverse document frequency: G(i) = log(N/n(i)) where N is the total number of sentences in the document, and n(i) is the number of sentences that contain term i.

When the weighted term-frequency vector Ak of a sentence k is created using one of the above local and global weighting schemes, we further have the choice of:

* Normalization: which normalizes Ak by its length |Ak|.
* No normalization: which uses Ak with its original form.

Therefore, for creating vector Ak of a sentence k, we have a total of 4 x 2 x 2 = 16 combinations of the possible weighting schemes. In our experimental evaluations, we have studied nine common weighting schemes, and their performances are shown in Figure 1. As seen from the figure, summarizer 1 is less sensitive than summarizer 2 to the changes of weighting schemes. Any of the three local weighting schemes (i.e. Binary, Augmented, logarithm) produces quite similar performance readings. Adding a global weighing and/or the vector normalization deteriorates the performance of summarizer 1 by 2 to 3% in average. In contrast, summarizer 2 reaches the best performance with the binary local weighting, no global weighing and no normalization (denoted as BNN) for most of the cases, while its performance drops a bit by adding the global weighing, and deteriorates dramatically by adding the normalization into the formula.



1. **Conclusion:**

In this project we build an application using two text summarization methods that create generic text summaries by ranking and extracting sentences from the original documents. The first method uses standard IR methods to rank sentence relevancies, while the second method uses the latent semantic analysis technique to identify semantically important sentences, for summary creations. Both methods strive to select sentences that are highly ranked and different from each other. We fetched the subtitles of a YouTube video and then we applied the two methods to get the summary. Every text, article, paragraph or document contains sentences that represent the main topic and some other sentences that represent supporting topics to our main topic. By generating the summary what we get is list of sentences that best represent the main topic in decreasing order. This is an attempt to create a summary with a wider coverage of the document's content and a less redundancy. Despite the very different approaches taken by the two summarizers, they both produced quite compatible performance scores. This fact suggests that the two approaches interpret each other. The evaluations also included the study of the influence of different VSM weighting schemes on the text summarization performances. Finally, the causes of the large disparities in the evaluators' manual summarization results were investigated, and discussions on human text summarization patterns were provided.

In future work, we plan to investigate machine learning techniques to incorporate additional features for the improvement of generic text summarization quality. The additional features we are currently considering include linguistic features such as discourse structure, anaphoric chains, etc, semantic features such as name entities, time, location information, etc. As part of the large-scale video content summarization project, we also plan to investigate how image and audio acoustic features extracted from video programs can help to improve the text summarization quality, and vice versa.

1. **Contributions:**

**Fahad Qureshi:**

* Implementing Latent Semantic Analysis.

**Talha Jamal:**

* Implementing Relevance Measure.

**Muhammad Naveed:**

* GUI.
* Testing & debugging results.
* Report Paper.

**References:**

[1] M. Sanderson, \Accurate user directed summarization from existing tools," in Proceedings of the 7'th international Conference on Information and Knowledge Management (CIKM98), 1998.

[2] B. Baldwin and T. Morton, \Dynamic coreference-based summarization," in Proceedings of the Third Conference on Empirical Methods in Natural Language Processing (EMNLP3), (Granada, Spain), June 1998.

[3] R. Barzilay and M. Elhadad, \Using lexical chains for text summarization," in Proceedings of the Workshop on Intelligent Scalable Text Summarization, (Madrid, Spain), Aug. 1997.

[4] C. Buckley and et al., \The smart/empire tipster ir system," in Proceedings of TIPSTER Phase III Workshop, 1999.

[5] J. Goldstain, M. kantrowitz, V. Mittal, and J. Carbonell, \Summarizing text documents: Sentence selection and evaluation metrics," in Proceedings of ACM SIGIR'99, (Berkeley, CA), Aug. 1999.

[6] E. Hovy and C. Lin, \Automated text summarization in summarist," in Proceedings of the TIPSTER Workshop, (Baltimore, MD), 1998.

[7] <http://www.SRA.com>.

[8] <https://www.aclweb.org/anthology/D18-1088/>

[9] <https://www.sciencedirect.com/science/article/abs/pii/S0306457304000329>