Latent Semantic Analyzer using Single Value Decomposition for Text Summarization

Krishnan Ravichandran
kxr200008@utdallas.edu

Raksheet Ramdas Bhat
rrr200001@utdallas.edu

Suchith Natraj Javali
sxj200024@utdallas.edu

***Abstract*—Text summarization is an efficient technique for producing a condensed summary of lengthy texts by focusing on the sections that provide relevant information without distorting the overal l meaning. By automating this process, we eliminate the challenge of turning lengthy documents into well-structured summaries, which might be challenging and expensive to complete if done manually. Text summarization is one of the widely researched problems in the Natural Language Processing(NLP) field and various methods for solving this problem emerge frequently. Latent Semantic Analysis(LSA) is one such technique for extractive text summarization. In this report, we apply LSA to the CNN newspaper articles dataset, and the algorithm's performance is measured using ROUGE scores. Our implementation of the LSA algorithm performs well as the SVD step (Singular Value Decomposition) of the LSA algorithm can learn the most important sentences in a given article. We also showcase the effectiveness of the summarization by calculating the reduction factor. Finally, we briefly discuss different evaluation metrics used to evaluate text summarization methods. We also present our ROUGE scores, compare them with similar implementations, and discuss the scope for future work on ways to better our implementation of the LSA algorithm.**

***Index Terms*—information retrieval; Latent Semantic Analysis; text summarization**

I. Introduction

Summarization in general is the process of condensing a long text into an acceptable amount while retaining all relevant information and the text's original meaning. Automatic content summarization has become very popular in the domain of academia because doing so manually takes a high amount of time and is typically challenging. The current expansion of non-structured textual data in the digital sphere necessitates the creation of automatic text summary technologies that make it simple for users to draw conclusions from them. These days, we have access to large volumes of information that are available publicly online.

However, a lot of this data is unnecessary, trivial, and might not convey the intended meaning. Therefore, it is becoming increasingly important to use automatic text summarizers that can extract useful information while excluding unnecessary data. Implementing summarizing can make information easier to read, cut down on the time spent looking for information, and make it possible to put more information into a given space.

Finding the information required to meet a user's demands is challenging due to the increase in documents that are now available online. From a given document, text summarization techniques extract key information. Any user can quickly discern if a document is relevant to their needs without having to read through the entire document, simply by reading the generated summary. The components of a summary are described as follows in the following research papers [1], [2]: first, one or more documents can be used to construct a summary. Second, a summary does not include extraneous material and includes only the information that is necessary. Third, a summary should be brief, or at least its length should be less than 50 percent of that of the source text.

Depending on how the summary is made, text summarizing systems can be classified as either extractive or abstractive system. Additionally, they can be further divided into single-document and multi-document summarizations depending on the count of input documents used. Generic or query-based summarizing is a different type of categorization that is based on the goal of summary. There are various methods based on supervised or unsupervised methodologies for summarizing systems in the literature. Beginning in the late 1950s, surface-level information served as the foundation for the earliest work on document summarization. Later, statistical techniques, more semantically focused analytic techniques like lexical chains, and algebraic-based techniques like Latent Semantic Analysis (LSA) for text summarization were created.

II. Background Work

Extractive and Abstractive summarizations are the two primary approaches to automatic text summarization. Using a predefined scoring algorithm, extractive summarization selects phrases from the source text to create a meaningful summary. This technique involves highlighting key passages in the text, clipping them out, and then piecing the content back together to create a streamlined version. Hence, this technique is entirely dependent on extracting phrases or sentences from the source text. As it is simpler and produces summaries that are naturally grammatical, extractive summarization has received the majority of attention in today's summarizing research.

On the other hand, abstractive summarization approaches seek to create summaries by analyzing the text using sophisticated natural language algorithms to build a new, shorter text that highlights the most crucial details from the original document. To produce summaries akin to what a human-written abstract generally produces, this calls for rephrasing sentences and adding information from the full text. Actually, a good abstractive summary is linguistically flexible and includes the

most important details from the input. So they are not restricted to simply choosing from the original text.

We decided to implement extractive summarization - and the technique we chose to do it with is Latent Semantic Analysis (LSA). Latent Semantic Analysis is a solid and effective Algebric-Statistical technique that extracts the features that cannot be explicitly specified from words and sentences by revealing their underlying semantic structures. Although not original to the dataset, these traits are crucial to the data. It is an unsupervised NLP technique.

We changed our algorithm from sequence to sequence model to LSA because after spending a lot of time researching seq2seq models, we realized that training Deep Learning models with pyspark was not feasible - Deep Learning support for pyspark is still very nascent. So we decided to use a different NLP technique to implement automatic text summarization.

III. Latent Semantic Analysis

Latent Semantic Analysis is an unsupervised method that requires no prior training or outside information. The context of the input material is used by LSA to extract information, such as the words that are frequently used together and in various phrases. Sentences with a lot of terms in common are likely to have a similar semantic meaning. The words in a sentence establish its meaning, and the sentences in which a word appears determine its meaning. It is possible to determine the relationships between phrases and words using the algebraic technique known as singular value decomposition. In addition to being able to model links between words and phrases, SVD can also reduce noise, which helps to increase accuracy.

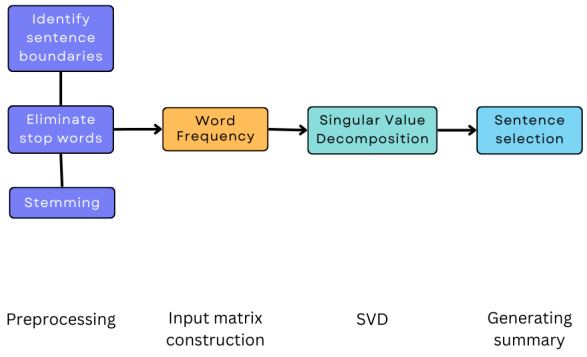

Fig. 1. LSA flowchart

A. Step I - Preprocessing

A series of preprocessing steps need to be applied to the original document to allow us to effectively perform LSA. The steps can be given as follows:

- Identifying sentence boundaries - Based on the language's punctuation used in the document, we break down the document into a list of sentences. Each sentence is treated as a separate entity.
- Eliminating stop words - Words that are frequently used but lack any meaningful semantics or information that is pertinent to the task are eliminated. To obtain a list of the most popular stopwords, we use the NLTK stopword list.
- Stemming - In order to emphasize the semantics of each word, we extract the stem or radix of each word. This allows to aggregate words with similar meanings, and replace every word with its root word.

B. Step II - Input matrix construction

The representation of an input document must allow a computer to understand it and do calculations on it. The typical form of this representation is a matrix, where the rows are words/phrases and the columns are sentences. The cells show the importance of each word in a sentence. The matrix that is produced is usually sparse since not all terms are utilized in every phrase. There are several methods for entering values into the cell.

The process of creating an input matrix is crucial for summarization because it affects the matrices that SVD employs to compute the results. The SVD algorithm has a high degree of complexity, which is made worse by the size of the input matrix, as was already established. By using techniques like removing stop words, restricting the use of words to their roots, replacing words with phrases, and other similar techniques, the rows of the matrix, or the words, can be made smaller. The results of the SVD can also be affected by the matrix's cell values. There are several ways to enter data into the cells. These approaches are as follows [3]:

- Frequency of word: The value of the cell represents the word's frequency in the sentence.
- Binary representation: the cell's value is 0 or 1 based on whether it occurs in the sentence or not
- TF-IDF (Term Frequency-Inverse Document Frequency): the word's TF-IDF value is the value of the cell. If a word appears more frequently in the given sentence but less frequently overall, the TF-IDF score will be higher.
- Log entropy: The log entropy of the supplied word serves as the cell's value. This tells us how much new information the word brings to the phrase.
- Root type: If the word's root type is a noun, the cell's value is its frequency; otherwise, its value is taken to be 0.
- Modified TF-IDF: The word's TF-IDF value is entered into the column, and if it is smaller than the row's average, it is replaced with 0.

C. Step III - Singular Value Decomposition

SVD is an algebraic technique for describing relationships between phrases, words, and sentences. As part of this method, the given input matrix is decomposed into 3 new matrices

$$A = U\Sigma V$$

A is the input matrix, U is a (words × extracted concepts) matrix, Σ is a diagonal matrix containing singular values and V is a (sentences × extracted concepts) matrix. A detailed diagram is given below [4]

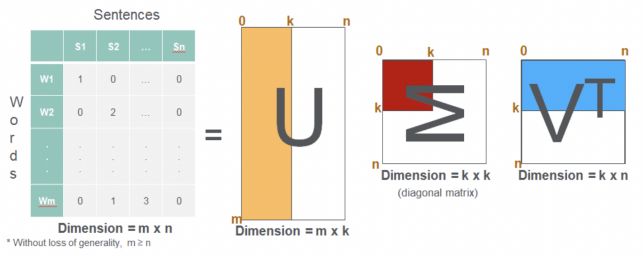

Fig. 2. SVD explained

SVD can be interpreted as follows, in the context of the given text summarization task:

- Maps an m-dimensional space to an r-dimensional singular vector space, where r = rank(A)
- Establishes r linearly independent foundation concepts for the document (or vectors)
- By identifying prominent and recurrent patterns in the document-term matrix and modeling them using one of the singular vectors, SVD can semantically group words and sentences together.
- If the pattern that the corresponding singular vector represents in the given document is higher, then its value is also higher.
- The sentence with the largest index value represents this pattern most effectively, hence we pick sentences with the largest index value.
- Finally, we select the sentences with the largest index values as summaries.

D. Step IV - Sentence selection

After running SVD, we use an existing algorithm [5] to choose the best sentences for the summary (sentences are selected because this is an extractive summarization type, so the summary is made up of sentences from the original document). In this technique, the most significant sentences are chosen using the Vt matrix from SVD.

The order of the rows in the Vt matrix indicates the relevance of the concepts, with the first row representing the most important concept that was retrieved. The cell values in this matrix shows how the sentence and the notion are related. The higher the cell value, the more closely the language relates to the concept.

IV. Results and Analysis

A. About the dataset

The CNN Newspaper/DailyMail Dataset in English is made up of original news articles written by journalists at CNN Enterprise and the Daily Mail Enterprise. The presented version now supports both extractive and abstractive summarization, despite the fact that the initial version was created for automated reading, comprehension, and abstractive question answering.

To provide data for our technique, we subsample 13,368 articles from the dataset at random. The structure of the dataset is as follows:

{ 'id': '...', 'article': '...', 'highlights': '...' }

This sample contains articles of various degrees of size. The largest document contains 1917 words and the smallest contains 41 words. The average size of the articles is 676 words.

B. Evaluation metric

It has historically been difficult to evaluate text summarizing techniques, and in the early stages of research, human interaction was necessary to fully assess a summarizing methodology. Even though numerous semi-automated techniques for evaluating text summarizing approaches emerged in the late 1990s, they were still inefficient for analyzing vast amounts of data. Scientists suggested utilizing cosine similarity to compare the model summary with the reference summary in the early 2000s.

Although this approach was quite successful, ROUGE (Recall-Oriented Understudy for Gisting Evaluation) eventually supplanted it as the most reliable way to assess automatically generated summaries. Based on the premise that two texts have comparable meanings if they share words and phrases, this evaluation metric, which was inspired by BLEU, was developed.

Given this context, we decided to use the ROUGE metric to assess the summaries that our algorithm produced. There are various variations of this metric, all of which grade the automatically generated summary numerically by contrasting it with the reference summary. Our model produced summaries, and to assess the performance, we computed ROUGE-N (1) and ROUGE-L scores.

ROUGE-1 will calculate this ratio using unigrams taken from both summaries, while ROUGE-N takes into account common n-grams between the auto-generated summary and the summary. We determine the precision, recall, and F-measure using these parameters.

$$Precision = \frac{\#of\ matching\ words}{\#\ words\ in\ generated\ summary}$$

$$Recall = \frac{\#\ of\ matching\ words}{\#\ words\ in\ reference\ summary}$$

$$F - measure = 2\,\frac{Precision\ *\ recall}{Precision\ +\ recall}$$

Similar to this, the ROUGE-L calculates the aforementioned statistics by determining the LCS, or longest common

sub-sequence, between the reference summary and the auto-generated summary. For this data set, we compute the ROUGE values using a single reference summary even though the ROUGE measure also permits evaluation with multiple reference summaries.

The CNN/Daily Mail data set comes with a label named "highlights". By using this as the reference summary in the above formulae, we obtain the following results:

Metric	Precision	Recall	F-measure
ROUGE-1	0.38399	0.27019	0.30565
ROUGE-L	0.22603	0.15667	0.17816

Since the summaries are generated for multiple sample texts the ROUGE scores are averaged.

C. Interpretation of the results

The primary object of a text summarization problem is to cut down large sections of the given text and to present it in a human-readable concise format. With respect to the primary objective, our model has achieved its purpose by cutting down huge volumes of sentences. We limited the model to selecting 3 sentences or fewer from the input text. This allowed the model to select a summary that only comprises 12% of the original content. The ratio of the words in the original text to those in the summary is known as the reduction factor. Note that the dataset contains articles of varying lengths, with the smallest of them containing just 41 words.

Nearly 40% of the words in the generated summary match those in the reference summary, according to the ROUGE score table above. We can infer from the recall scores that around 30% of the reference summary corresponds to the created summary. The F-measure, which accounts for both recall and precision, is 30%. Given that the reference summary is an abstractive summary of the document and word matches are rather few, these results are competitive with the top extractive summarization algorithms.

The sentences in the abstract-based summary in the "highlights" column were constructed based on the knowledge of the article and were not directly taken from the article, which causes a tiny decline in matching scores for ROUGE-L scores. Our program produced summaries with a 25% sequence match even for sentences that were not taken directly from the text.

Note that the ROUGE score does not assess the fluency of the generated summary. ROUGE simply measures adequacy by counting the matching n-grams or LCS between the generated summary and the data set's summary.

D. Comparison with other implementations

Our model performed on par with other comparable methods that were performing similar extractive text summarization.

Method	GI	SJ	MRC	Cross	Topic
freq	0.236	0.250	0.244	0.302	0.244
tf-idf	0.200	0.218	0.213	0.304	0.213

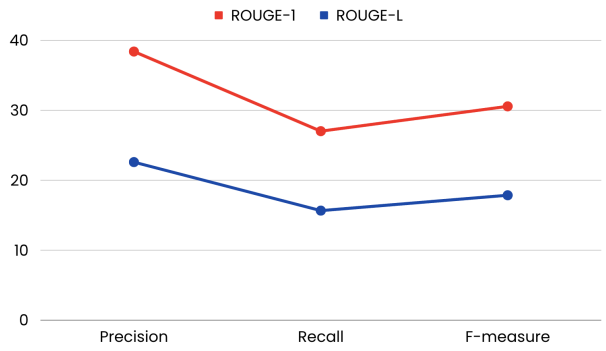

Fig. 3. ROUGE Scores

The above table showcases the ROUGE-L scores about extractive summaries performed with different algorithms on a similar data set. GI (Gong and Liu), SJ, MRC, Cross, and Topic are various ways to derive important terms and sentence selections from the document.

Our implementation of SVD with Gong and Liu (GI in the above table) outperformed the implementation published in this article [3]. We also observe that every extractive summarization method performed poorly compared to Machine Learning models which mostly does abstractive summary. The summaries from ML models rely on generating new sentences that are humanly understandable in a concise format, hence producing higher ROUGE scores.

V. Conclusion and Future Work

Finding information that is relevant to a user's needs among the many publications has gotten harder with the rise of text-based resources. Techniques for text summarizing are recommended and put to the test as a solution to this problem. The extraction of simple features was the starting point for the study of summarization, which advanced to the use of numerous techniques, such as lexical chains, statistical techniques, graph-based techniques, and algebraic solutions. Latent Semantic Analysis is one of the statistical and algebraic techniques.

The CNN dataset was used to run the LSA algorithm, and ROUGE scores were used to evaluate the results of the algorithm. Our algorithm performed on par (ROUGE-1 and ROUGE-L metrics) with similar implementations of LSA [3].

In the future, there are several other ideas to fill the cell values of the document-term matrix. We computed the term frequencies of each word, but there are other ways like the TF-IDF value of each word, log entropy, root type, etc. We'd like to explore other ways to compute the document-term matrix and compare the results to see which method performs the best.

REFERENCES

[1] Radev DR, Hovy E, McKeown K. Introduction to the special issue on summarization. Computational Linguistics 2002; 28(4): 399–408.

[2] Das D, Martins AFT. A Survey on automatic text summarization. Unpublished manuscript, Literature survey for Language and Statistics II, Carnegie Mellon University, 2007.

[3] Makbule Gulcin Ozsoy and Ferda Nur Alpaslan, Ilyas Cicekli, "Text summarization using Latent Semantic Analysis" Journal of Information Science, August 2011.

[4] Blog: Text summarization using singular value decomposition, https://bicorner.com/2016/03/28/text-summarization-using-single-value-decomposition

[5] Gong Y, Liu X. Generic text summarization using relevance measure and latent semantic analysis. In: Proceedings of SIGIR '01 2001

[6] Orasan, Constantin. "Automatic summarisation: 25 years On." Natural Language Engineering 25 (2019): 735 - 751.

[7] Blog: An intro to ROUGE, and how to use it to evaluate summaries, https://www.freecodecamp.org/news/what-is-rouge-and-how-it-works-for-evaluation-of-summaries-e059fb8ac840/

[8] Blog: The Ultimate Performance Metric in NLP, https://towardsdatascience.com/the-ultimate-performance-metric-in-nlp-111df6c64460

[9] Josef Steinberger, Karel Ježek, Using Latent Semantic Analysis in Text Summarization and Summary Evaluation

[10] K. Merchant and Y. Pande, "NLP Based Latent Semantic Analysis for Legal Text Summarization," 2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI), 2018, pp. 1803-1807.

[11] O. Tas and F. Kiyani, "A Survey Automatic text Summarization", PressAcademia Procedia, vol. 5, no. 1, pp. 205-213, Jun. 2017, doi:10.17261/Pressacademia.2017.591

[12] Mani, I., Klein, G., House, D., Hirschman, L., Firmin, T., and Sundheim, B. (2002). SUMMAC: A text summarization evaluation. Natural Language Engineering, 8(1), 43-68.

[13] M. Bhandari, P. N. Gour, A. Ashfaq, P. Liu, G. Neubig, 'Re-evaluating Evaluation in Text Summarization', CoRR. abs/2010.07100, 2020.