NLP Based Latent Semantic Analysis for Legal Text Summarization

Kaiz Merchant
B.E in Computer Engineering
Dwarkadas J. Sanghvi College of Engineering
Mumbai, India
merchant.kaiz@gmail.com

Yash Pande
B.E in Computer Engineering
Dwarkadas J. Sanghvi College of Engineering
Mumbai, India
yash3096@gmail.com

***Abstract*— It is very essential for lawyers and ordinary citizens to do an exhaustive research related to their case before they answer questions in court. For quite some time they have had to read extremely long judgements and try to pick out the useful information from them or hire legal editors to create summaries. We propose an automated text summarization system that generates short and useful summaries from lengthy judgements. We make use of a natural language processing technique called latent semantic analysis (LSA) to capture concepts within a single document. We use two approaches- a single document untrained approach and a multi-document trained approach depending on the type of input case (criminal or civil). Our data was collected from official government sites that included Supreme Court, high court and district court cases and our model achieved an average ROGUE-1 score of 0.58. Finally, our system was approved by professional lawyers. In the future we aim to provide better continuity within our generated summaries and evaluate our system more accurately.**

Keywords— Latent semantic analysis, legal, text summarization, natural language processing

I. INTRODUCTION

Legal text processing is a very important issue in today's world due to the large amount of information available in the domain. Extracting only the useful parts from the large amount of data proves to be of great importance not only to lawyers but also to ordinary citizens. They often need to answer many questions relating to a case and also refer to previous cases that are similar to theirs in order to get a better idea and deeper understanding of the situation. However, the legal judgements are usually very long and may consist of a number of pages. Reading through all this information is a very tedious task. Thus, legal editors are often hired to prepare what we call as human generated summaries. However, editing so many cases takes a lot of time and does not guarantee a dependable outcome. To solve this issue, we need a system that can automatically generate summaries from the given cases. This is possible through automated text summarization. This process aims to create short summaries from the lengthy text without changing the main idea of the original document. It also ensures that all the important points are included in the summary. Sometimes, these shorter representations can be used to assist the human editors.

Many text summarization systems have been developed over the years. Among these, the most recent and successful approaches use deep learning [1]. However, deep learning depends purely on previous data and does not consider semantics within a single new document. For legal text summarization, every case seems to be different and thus relying on already seen data is not a good approach. Hence, in our proposed system, we make use of latent semantic analysis that considers similar concepts within a piece of text. Another issue with legal data is that criminal judgements differ in their pattern from civil ones. While each civil case is different from the other, criminal cases tend to have a kind of similar flow. Thus we make use of two approaches, a single document approach and a multi document approach.

II. RELATED WORK

There have been various approaches to text summarization over the years. Luhn HP [2] used relative positioning and frequency of the words for selecting sentences and create summaries out of it. He did not consider concept modelling and other semantic relationships among sentences. Kupiec J et al. [3] used a Bayesian classification function to find probabilities of all sentences and rank them in hand-picked document extracts. Scoring every sentence of lengthy legal documents will be computationally bad. Shashi Pal Singh et al. [4] proposed an extractive unsupervised Deep Learning approach with a two hidden layer Restricted Boltzmann Machine for sentence extraction. The first sentence was always included, while the top 50% sentences from the descending order score sorted list were taken and their cosine similarity was calculated repeatedly. K. Kaikhah [5] used a three layer feed-forward neural network to discover patterns existing in the sentences that were needed to be included in the summaries. Every sentence was chosen manually. Moreover, this approach disregards certain features by pruning the neural network which might not give proper summaries for legal documents as many important features might not be included.

Various statistical algorithms and semantic based analysis like lexical chains [6,7] and algebraic-based approaches[8] have also been used previously for generating summaries. Gonenc Ercan et al. [7] used lexical chains to analyse

978-1-5386-5314-2/18/$31.00 ©2018 IEEE

linguistic cohesion among text and then extract significant sentences using semantic relationships. In both these methodologies WordNet and dictionaries were used extensively to identify synonyms. These can work well for criminal legal documents having detailed stories but not for the civil ones.

Out of the two approaches widely followed automated extractive text summarization is easier to implement than automated abstractive text summarization because the latter has a complex procedure of generating new sentences from the original documents based on their understanding. LSA proposed by Thomas K Landauer et al. [9] is one such extractive statistical summarization technique widely used. Josef Steinberger et al. [10] introduced two new evaluation strategies based on LSA. The modified LSA algorithm found the most significant topics out of the document. The matrices U and V were multiplied with their corresponding singular values after the SVD process and sentences with highest values were chosen. This enhanced the overall LSA summarization and evaluation process as the dimensions which had singular values in the higher 50% of the total number of dimensions were given preference. Although they compared seven summaries by a content-based evaluator like cosine similarity and 2 new LSA evaluators, their results could have gone down for lengthy documents with many important dimensions. Ozsoy et al. [11] proposed two summarization algorithms based on extractive LSA algorithms which were evaluated using ROGUE scores on Turkish and English documents. They used datasets from different realms like medicine, psychology, computer science and sociology for evaluation based on different conditions and achieved impressive results. However, they could have modified their model for different type of documents after analysing the input data to improve evaluation results. Rasha Mohammed Badry et al. [12] also used the traditional LSA algorithm for text summarization and performed a comparative study of all the existing LSA algorithms. Since SVD does not perform well for new documents due to complex calculations, their model could have been improved. I. V. Mashechkin et al. [13] used LSA for text summarization along with NMF (Non-negative Matrix Factorization) to measure the relevance of sentences. Throughout NMF each topic was weighed with the help of sentence portrayal in normalized topic space and thus the internal structure of text was better preserved. Oi-Mean Foong et al. [14] gives us another example in which text summarization is implemented in android platform using LSA with a slightly modified calculations of weight of words and sentence score.

Although all these works have used DUC 2001, 2002 and 2004(Document Understanding Conferences) datasets for assessing their models, none of them found results for implementing their model on legal documents which are organised in a completely different way.

III. Dataset

Our data consists of legal judgements issued by the Indian judiciary system. Supreme court, high court and district court data was collected from official sites having extensions of .nic, .gov, etc. To retrieve this data, a web crawler was created in python using the Selenium[15] and Beautiful Soup [16] libraries. This method of gathering data is referred to as web scraping. More than 1 lakh documents were collected from which a group of 50 were selected, consisting of documents from all kinds of courts and a variety of sources. This group of documents also contained judgements of both types- civil as well as criminal. This group formed our dataset.

IV. Proposed System

The entire flow of the system is depicted in Fig. 1.

A. Pre Processing

The first step in pre-processing is to consider only the judgement part of the input document. The input document is an xml file consisting of fields such as petitioner name, respondent name, cases cited, headnote and judgement. The text inside the judgement tag is retrieved using an xml parser in python. The next step is to remove punctuations that are part of abbreviations such as “Mr.” because they might mislead the model into considering wrong ends of statements.

This is successfully done by means of using regular expressions. Finally, an approach called stemming [17] is applied which transform words into their root words so that they can be considered as identical during evaluation. For example, “killing” may be changed to “kill”. Once pre-processing is performed, the document can be passed to the model.

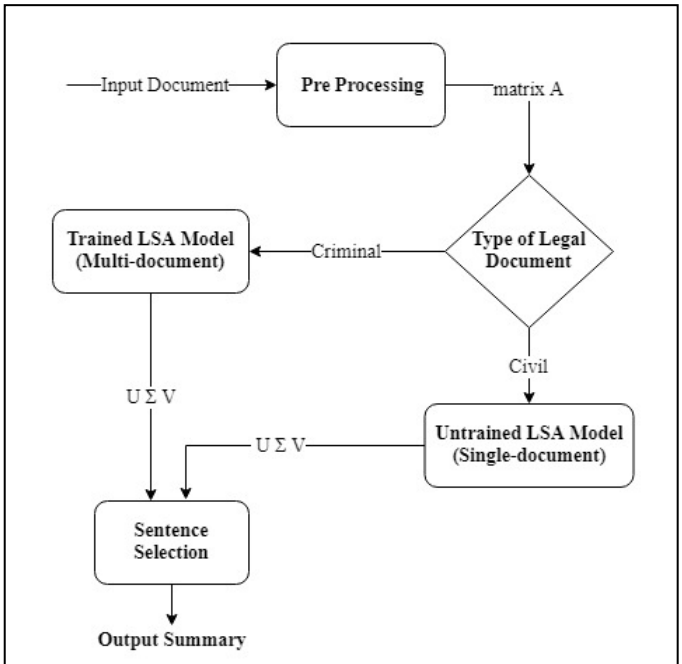

Fig. 1. Flow of the System

B. Latent Semantic Analysis(LSA) Model

Latent Semantic Analysis [14] also known as Latent Semantic Indexing (LSI) is a completely automated unsupervised statistical-algebraic summarization technique which follows an extractive approach towards analyzing documents and finding hidden semantic relations among words and sentences of the text. LSA is widely used in the field of natural language processing for information retrieval and feature space expansion in text mining, document comparison in low-dimensional space, and for finding semantic similarity between terms and documents. LSA assumes that semantically similar terms will occur in similar pieces of text. It analyzes the relation between a set of terms and documents containing them by producing a set of concepts. Every document is represented as a two-dimensional matrix A having mxn dimensions, where m is the number of terms and n is the number of sentences in the preprocessed input text document. Internally, LSA uses singular value decomposition (SVD) to reduce the number of rows of this matrix by performing cosine similarity comparison of words and reducing noise.

a) Singular Value Decomposition(SVD): The values in the input matrix A indicates the importance of words with respect to each sentence. The input text is preprocessed to reduce the dimensionality of A matrix as the complexity of SVD is directly affected by its size. Of all the different approaches for filling the cell values of the matrix A [2] like binary representation, Tf-idf, log entropy, root type, modified Tf-idf, we use the frequency of words occurring in sentences. Since most of the words don't occur in more than few sentences A is a sparse matrix with very less non-zero values. SVD decomposes A into three new matrices U, V and Σ according to the formula

$$A = U \Sigma V^* \quad (1)$$

U is a m x r matrix which stores left singular vectors, where m represents words and r represents concepts or topics.
Σ is a r x r special non-negative diagonal matrix of singular values giving the strength of concepts sorted in descending order. It has zeroes everywhere except the diagonal.
V is a n x r matrix which stores right singular values, where n is the number of sentences in a document and r represents the concepts discovered. V* is the conjugate of V.

U and V are going to be orthogonal matrices and are also called column orthonormal matrices. This means that their columns have Euclidean length one i.e the sum of squared values of each column of U and V equals one and the dot product of any two columns of U or V equals zero. The values of U, Σ and V are calculated using the following two equations:

$$A^* A = V \Sigma^* \Sigma V^* \quad (2)$$

$$A V = U \Sigma \quad (3)$$

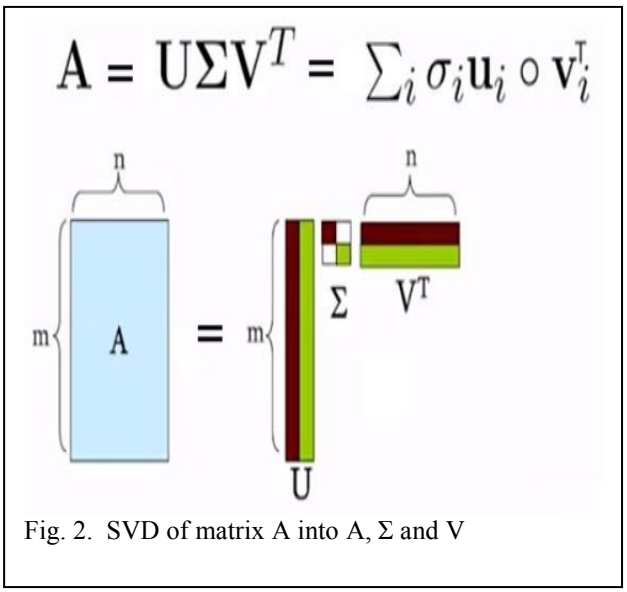

Fig. 2. SVD of matrix A into A, Σ and V

From the diagonalization of A*A (2), eigen values and corresponding eigen vectors are found. Eigen values of A*A are the squared values of diagonal entries of Σ and the eigen vectors are the columns of V. Thus any input matrix of a document is decomposed in a unique set of matrices U, V and Σ in order to discover concepts.

We have improved our system by using two different approaches, one for civil and the other for criminal cases. Criminal judgements have a similar flow and many times use identical terms. Thus we implemented a multi-document trained approach for them, as scores previously generated from SVD of multiple documents had an impact on newly generated ones. Civil judgements on the other hand very rarely have repeated terms and ideas. Therefore we implemented an untrained LSA model approach where the word score of every civil judgement input matrix after SVD was not preserved and for all civil documents it was independent of any previously produced scores.

C. Sentence Selection

After the LSA model completes its analysis, the top sentences are picked for the final summary. The length of the summary was restricted to 5 percent of the original length which is short enough for people to read. Apart from including the sentences selected by the LSA model, we unconditionally add the last sentence. This sentence usually conveys the final judgement such as "Appeal dismissed". This is an important point and usually present in the reference summary. Adding this sentence thus helped improve the overall accuracy of our system.

V. Results

Our summarization model was successfully implemented and the results were generated.

a) Experimental Setup: Two summarization models were built using python. One of them used training by allowing multi-document criminal summaries to be passed to a single program with a single execution. The other model iteratively passed civil summaries to the program with a new execution for each document. The directories containing the judgements and reference summaries were given as input and the system generated summaries were saved to a new directory in the form of the output. The ROGUE scores were displayed on the screen.

b) System Evaluation: The system evaluation is depicted by an example sentence-

System generated: "The Trial Court found all the accused guilty of the charges and convicted and sentenced them."

Reference: "Accused found guilty and sentenced to jail."

The results were partially evaluated using Recall-Oriented Understudy for Gisting Evaluation (ROGUE) [18].
This evaluation compares system generated summaries with reference or human generated summaries. If more than one references exist, average score is used. The variations of ROGUE used for evaluation of our system include ROGUE-1 (which is based on overlap of a single word (1-gram) between the system and reference summaries), ROGUE-2 (overlap of bigrams), and Rogue-L (using longest common subsequence). The average scores obtained for each type of test are presented in Table. I. below.

However, the effectiveness of the system cannot be completely judged by the ROGUE scores. This is because the test takes specific words into account. The reference summaries have been written by expert lawyers and then modified by professional editors and thus there is a tendency for the words in the machine generated summaries to not match those in the human generated ones. Also, there is a great length difference between the system and human generated summaries due to one being extractive and considering whole sentences while the other being abstractive and trying to inculcate many concepts into the same sentence. Considering these facts, the ROGUE-1 score that we achieved is more than acceptable. Also, when evaluated carefully by renowned lawyers, the ideas in both the summaries were almost completely matching. This provides evidence of the success of our model.

TABLE I. Test Results

Type of Test	Average Score Obtained
ROGUE-1	0.58
ROGUE-2	0.15
ROGUE-L	0.35

VI. Conclusion and Future Work

We successfully implemented a legal text summarizer using our proposed model which makes use of a natural language processing technique called latent semantic analysis. A short summary was generated keeping intact the important ideas from the original document. We were able to achieve an average ROGUE-1 score of 0.58. However, this evaluation method is not completely effective and in the future we aim to evaluate our model based on entire concepts and not only similar words. Furthermore, using the LSA technique sometimes leads to a break in the overall continuity and we aim to modify our proposed system to improve on this issue. Finally, we aim to deploy our system on the mobile platform to increase its usability.

References

[1] K. Kaikhah, "Automatic text summarization with neural networks," *Intelligent Systems, 2004. Proceedings. 2004 2nd International IEEE Conference*, 2004, pp. 40-44 Vol.1. doi: 10.1109/IS.2004.1344634.

[2] H. P. Luhn, "The Automatic Creation of Literature Abstracts," in *IBM Journal of Research and Development*, vol. 2, no. 2, pp. 159-165, Apr. 1958. doi: 10.1147/rd.22.0159.

[3] Kupiec J, Pedersen JO, Chen F. A trainable document summarizer. Research and Development in Information Retrieval 1995: 68–73.

[4] S. P. Singh, A. Kumar, A. Mangal and S. Singhal, "Bilingual automatic text summarization using unsupervised deep learning," *2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT)*, Chennai, 2016, pp. 1195-1200.doi: 10.1109/ICEEOT.2016.7754874.

[5] K. Kaikhah, "Automatic text summarization with neural networks," *Intelligent Systems, 2004. Proceedings. 2004 2nd International IEEE Conference*, 2004, pp. 40-44 Vol.1. doi:10.1109/IS.2004.1344634.

[6] Barzilay R, Elhadad M. Using lexical chains for text summarization. In: Proceedings of the ACL/EACL '97 workshop on intelligent scalable text summarization 1997: 10–17.

[7] Ercan G., Cicekli I. (2008) Lexical Cohesion Based Topic Modeling for Summarization. In: Gelbukh A. (eds) Computational Linguistics and Intelligent Text Processing. CICLing 2008. Lecture Notes in Computer Science, vol 4919. Springer, Berlin, Heidelberg.

[8] Kadhar Batcha, Nowshath & Aziz, N.A.. (2014). An Algebraic Approach for Sentence Based Feature Extraction Applied for Automatic Text Summarization. Advanced Science Letters. 20. 139-143. 10.1166/asl.2014.5258.

[9] T. K Landauer, P. W. Foltz, and D. Laham "An Introduction to Latent Semantic Analysis" Discourse Processes, col. 25, pp. 259-284, 1998

[10] Josef Steinberger and Karel Jezek, "Using Latent Semantic Analysis in Text Summarization and Summary Evaluation", Proceedings of the 7th International Conference ISIM, 2004.

[11] Makbule Gulcin Ozsoy, Ferda Nur Alpaslan, and Ilyas Cicekli, " Text summarization using Latent Semantic Analysis", Journal of Information Science (2011), Vol 37, Issue 4, pp. 405 - 417 doi: 10.1177/0165551511408848.

[12] Badry, Rasha Mohammed, Ahmed Sharaf Eldin and Doaa Saad Elzanfally. "Text Summarization within the Latent Semantic Analysis Framework: Comparative Study." (2013).

[13] Mashechkin, I.V., Petrovskiy, M.I., Popov, D.S. et al. Program Comput Soft (2011) 37: 299. https://doi.org/10.1134/S0361768811060041.

[14] O. M. Foong, S. P. Yong and F. A. Jaid, "Text Summarization Using Latent Semantic Analysis Model in Mobile Android Platform," *2015 9th Asia Modelling Symposium (AMS)*, Kuala Lumpur, 2015, pp. 35-39. doi: 10.1109/AMS.2015.15.

[15] Full documentation for selenium library available at:https://www.seleniumhq.org/.

[16] Full documentation for Beautiful Soup library available at: https://www.crummy.com/software/BeautifulSoup/bs4/doc/.

[17] Ms. Anjali Ganesh Jivani, A Comparative Study of Stemming Algorithms, Anjali Ganesh Jivani et al, Int. J. Comp. Tech. Appl., Vol 2 (6), 1930-1938, ISSN:2229-6093.

[18] Lin.C.Y.2004.ROUGE: a Package for Automatic Evaluation of Summaries.Workshop on Text Summarization Branches Out (WAS 2004). 25-26.