2022 25th International Conference on Computer and Information Technology (ICCIT), 17-19 December, Cox's Bazar, Bangladesh

An Intelligent Approach Towards Legal Text-Documents Retrieval

1st Md. Mushfiqur Rahman
Department of CSE
Ahsanullah University of Science and Technology
Dhaka, Bangladesh
nishatmn.cse@gmail.com

2nd Zahin Azmaeen
Department of CSE
Ahsanullah University of Science and Technology
Dhaka, Bangladesh
azmaeen98@gmail.com

3rd Mithila Arman
Department of CSE
Ahsanullah University of Science and Technology
Dhaka, Bangladesh
mithilaarman30@gmail.com

4th Md Latifur Rahman
Department of CSE
Ahsanullah University of Science and Technology
Dhaka, Bangladesh
lrlemon147@gmail.com

5th Md Moinul Hoque
Associate Professor, Department of CSE
Ahsanullah University of Science and Technology
Dhaka, Bangladesh
moinul@aust.edu

2022 25th International Conference on Computer and Information Technology (ICCIT) | 979-8-3503-4602-2/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICCIT57492.2022.10054858

***Abstract*—Document retrieval is the process of discovering a set of documents related to a query or another document, as per comparable similarity to some significant extent. Such as legal law, cases, or judgments. The current work under the area of the Document retrieval domain focuses on creating a system that will examine various features or information contained in the supporting documents with the given query document and suggest the possible top-ranked documents which have a higher similarity score. This work is particularly useful in helping out the lawyers by suggesting similar cases or judgments and allowing them to explore significantly less number of previously recorded cases/judgments. The main focus of this work is to retrieve relevant documents for particular Supreme Court cases in India from a set of prior case documents. To categorize documents and do the document retrieval task, we have used a published data set of 2000 prior cases. For each distinct document, the current system lists a set of documents with their ranking scores using N_similarity, Fast-text, and BERT. Then, the system returns the relevant documents based on their ranking score. In order to compare our system to other similar ones already in use, we computed the accuracy and recall of our system. We have discovered that various methods perform better in certain areas where a few systems have higher precision and others with higher recall values. Our developed system provides comparably better accuracy, recall, and mean average precision than other current systems.**

***Index Terms*—information Retrieval, legal document, natural language processing, document similarity**

I. Introduction

In general situations, lawyers compare new cases to past ones before working on them. To solve a new case, they require to conduct numerous case studies. Additionally, several situations make it very challenging to compare a new caustic match to the prior cases. This work from a collection of earlier case materials selects the pertinent or citeable documents for certain Supreme Court cases. An Information Retrieval (IR) system searches a library of natural language texts in order to precisely retrieve a set of relevant documents that respond to a user's query. The relevant documents are those that fulfill the user's request. If the IR system is flawless, it will only retrieve pertinent documents. It is quite a challenging task to create a system that reliably understands the meaning of a text since human language is ambiguous. Programmers require to teach natural language-driven applications to recognize and understand accurately from the beginning if those applications are to be useful. By deconstructing human text and speech input in ways that the computer can understand, several Natural Language Processing techniques assist the machine in comprehending what it is taking in.

A. State-of-the-art

There are quite many works that perform similar tasks. The authors of the working team named flt_iela [1] from the Queensland University of Technology, Australia uses a fully automatic method where for each of the query documents they have formed a set of queries from the positions where the actual citations were present. They used BM25[2] algorithm for the prior case retrieval process.

The authors of the working team named LJIT2017_IRLeD_Task2 from the Heilongjiang Institute of Technology, China has used a language model based on Dirichlet Prior Smoothing [3], BM25[2] and Lucene[4].

The authors of the working team named SSN_NLP from the College of Engineering, India has used a Term Frequency-Inverse Document Frequency [5] to compare a legal document with precedent cases. Their system operates by escalating according to the frequency with which a word appears in a document but is counterbalanced by the number of documents in which the word appears. While considering the TF-IDF vectors they have considered only the nouns in the document.

The authors of the working team named rightstep-spune_1_task2 from the RightSteps Consultancy, India have used a weighted average of three different methods namely Regular Expression [6] based, Topic Modeling based[7], and Using Document Vector[8]. They calculated a weighted average of three distinct techniques to get the similarity score [9] between two cases: based on regular expressions Here, pattern matching was used to identify several legal statutes (such as Articles) that were mentioned in the text. The same attempt is made for every previous case after the list of statutes for a particular query document has been acquired. All previous instances with shared statutes are retrieved. Using topic modeling They use the gensim packages Latent Dirichlet Allocation (LDA) implementation for this strategy. Therefore, a similarity score is determined based on the proportion of topic terms that match the input to the total.

The team UB_Botswana_Legal_Task2 has done basic query formulation. For this, they have tokenized the text and removed all stopwords, and stemmed them using Porter Stemmer.

The team bphcTask2IRLeD from Birla Institute of Technology and Sciences, Pilani, India has considered a minimized set of words by considering only 5000 such words whose combined score of POS occurrence probability and IDF score is higher than the rest of the words. The similarity score between two document vectors is measured by simply finding the dot product of the two. For each Query Case, DPH-DFR [10], Bol model, and sequential dependence [11] are used in UB-Botswana-Legal-Task 2. In order to score and rank the prior examples, they used the formulated queries to deploy the parameter-free DPH term weighting model from the IR platforms Divergence from Randomness (DFR) framework.

The team named AMRITA-CEN-NLP-RBG used only the Doc2Vec method. They have used the Doc2Vec algorithm as implemented in the gensim package of python. Once the vectors are obtained the similarity between a query case document and a prior case is simply calculated as the cosine similarity between the two vectors. The top-ranked prior cases are reported for each of the current cases.

II. TASK

A. Dataset

There were two sets of documents (www.isical.ac.in) available for this task:

- *Current cases*: A collection of 200 Indian Supreme Court cases from which prior cases must be identified.
- *Prior cases*: For every current case, we gathered a list of prior cases that the case decision cited. The second collection of documents included 1000 more documents that weren't cited from any of the 1000 prior cases that were present in the "current cases" group.

In order to generate a ranked list of documents from the second set (prior cases) for each document *d* in the first set (current cases), we order the documents so that those that were genuinely cited from document d are ranked higher than the other documents.

B. A snippet of the Dataset

Summary of Current Case 1

Judgement IN THE SUPREME COURT OF INDIA CIVIL 2\. This appeal is directed against the judgment dated .

3\. The appellants are aggrieved by the impugned judgment of the High Court .. jurisdiction under Article 227 of the Constitution of India.

Summary of Prior Case 1256

227 of the Constitution. Code of Civil Procedure . move High Court under Article 227.

On the failure of the judgment.

The petitioner High Court under Art 227 of the Constitution contending that his death.

The High Court declined to .. .. Judge on various grounds.

On the question whether . a petition under Art 227 of the Constitution.

III. Proposed Model

Figure 1 shows the steps of the proposed model which contains several steps before ranking the prior cases compared to the current one.

Step 1: Pre-processing the dataset that contains prior cases and current cases: In this step, we load and read our dataset, Sort all the files. then read the current and prior cases files and encode it.

Step 2: This step involves cleaning the dataset and removing stop words and converting all content to lowercase for both the prior and current cases for embedding. We then convert them to lowercase and split them.

Step 3: Converting all the content of the corpus from upper case to lower for testing.

A. ***Model Training:***

After preprocessing, we trained five different models. Those are:

1. Sentence similarity using n_similarity: Here the sentence similarity is normally calculated by obtaining the embedding of the sentences and taking the distances between them. It is the task of determining how similar two texts are. This task is especially useful for information retrieval.

2. Sentence similarity using sorted n_similarity: This is quite similar to n_Similarity where sentence similarity

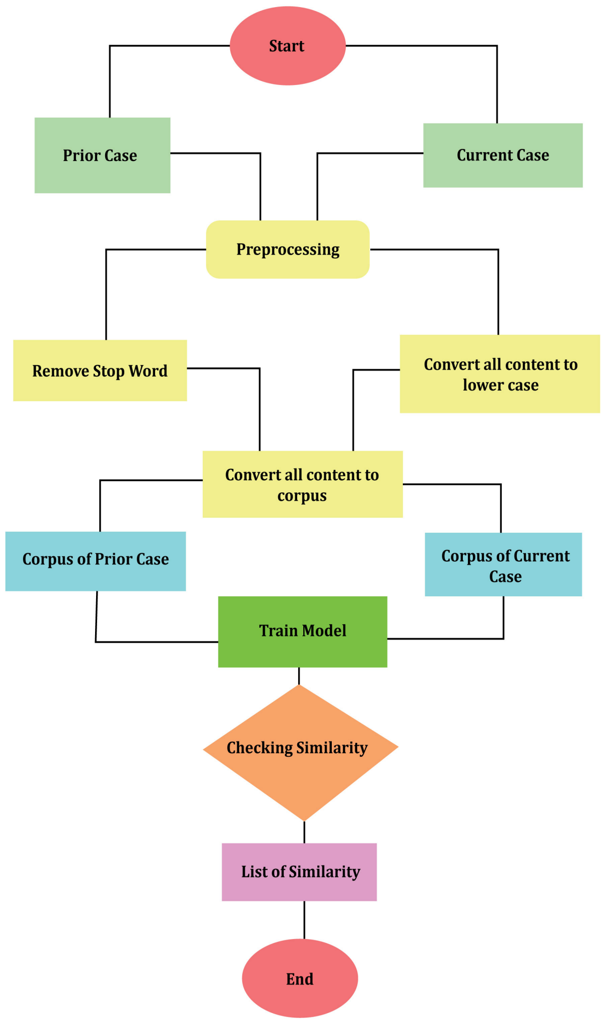

Fig. 1. Proposed Model

is obtained using the similarity between the sentences tokenized document and a sentence getting the result saved in a list. Then the results are sorted based on the similarity score.

3. Sentence similarity using wmdistance: Using this technique, we have meaningfully evaluated the ”distance” between two documents even when they do not share any words. Word2vec vector embeddings are used here. Although there are no words in common between the sentences, WMD may accurately gauge how similar or unlike they are by matching the pertinent words. The approachs underlying idea is to determine the shortest ”traveling distance” between documents, or the most effective way to ”transfer” the distribution of document 1 to the distribution of document 2, according to the method. Word Movers Distance calculates word embedding. Here, embeddings are mapped to n-dimensional space. We also know that words with similar words are closer in n-dimensional space.

For example- Inputsentence_ Obama = preprocess(Obama speaks to the media in Illinois) sentence_orange = preprocess(Oranges are my favorite fruit) distance = model.wmdistance(sentence_obama, sentence_orange) print(distance = Output: distance = 1.3663)

4. Sentence similarity using BERT-based model: Here sentence similarity is calculated using the Bert-based model. In BERT, the sentence is first converted to a vector. Next, turn a large number of additional sentences into vectors. Find the sentences that are closest to one another in terms of Euclidean distance or cosine similarity. then determine how semantically related the sentences are. It is just a Transformer language model with multiple layers of encoders and self-aware heads. With the release of BERT, anyone in the world can train their question-answering models, sentiment analysis, or other language models quickly and efficiently.

5. Sentence similarity using FastText:
This is another word embedding model we have used that builds on the word2vec concept named as FastText. FastText encodes each word as a Skip-gram of characters rather than learning word vectors directly. This helps us understand the meaning of short words and helps us understand embedding, suffixes, and prefixes. When a word is represented using the letter n-gram, a jump gram model is trained to learn to embed. This model does not take into account the internal structure of the word, so it is considered a word model with a sliding window above the word. The order of the n-grams doesn't matter as long as the characters are in this window.

IV. Experimental Verification

We have tested all five models for performance comparisons.

1) *Sentence similarity using N_similarity:* Sentence similarity was calculated by obtaining the embedding of the sentences and taking the distances between them.

- Run 01: First of all we used N_Similarity without tuning and preprocessing. After training 2000 prior cases, we tested 200 current cases and we got the following result.

TABLE I
N_Similarity Scores without tuning

MAP	Precision	Recall
0.2	0.119	0.552

- Run 02: In this run, we have removed stop words, using corpus, convert all the text context into lower for and test 2000 current cases. After that, we got a better result from the previous test. The result is presented below

- Run 03: We then tuned the model by changing the size of the epoch. Previously we tested with 10

TABLE II
SCORE OF N_SIMILARITY AFTER PREPROCESSING

MAP	Precision	Recall
0.2	0.14	0.62

epochs and afterward the model was tuned with 5 epochs. The result is presented below.

TABLE III
N_SIMILARITY SCORE AFTER TUNING THE MODEL

MAP	Precision	Recall
0.204	0.134	0.609

2) *Sentence similarity using sorted N_similarity:*
   - Run 01: This was similar to N_similarity. Here we sort the result on the basis of similarity. And we get the following outcome (Table 4)

TABLE IV
SCORE FOR THE SORTED N_SIMILARITY AFTER PRE-PROCESSING

MAP	Precision	Recall
0.306	0.17	0.804

   - Run 02: This run takes place after tuning the model by changing the epoch size. Previously we tested with 2 epochs and afterward we further tune the cases using 5 epochs and our result is presented in Table 5.

TABLE V
SORTED N_SIMILARITY SCORE AFTER TUNING THE MODEL

MAP	Precision	Recall
0.389	0.2009	0.976

   - Run 03: After running the two of the above processes, the second run (Sorted n_similarity after tuning) was found to be better than the rest of the runs. So we chose this run as our selected model.

3) *Sentence similarity using WM Distance:*
   - Run: We have used the WM Distance model and after training all the prior cases, we trained the current cases. Then we got the following result as shown in table VI.

TABLE VI
SCORE FOR THE WM DISTANCE

MAP	Precision	Recall
0.201	0.109	0.504

4) *Sentence similarity using Bert-base model:*
   - Run: In this run, we use the Bert Transformer Model. After training the prior cases, we tested all the current cases and we get the following result as shown in table VII.

TABLE VII
SCORE FOR THE BERT TRANSFORMER BASED MODEL

MAP	Precision	Recall
0.23	0.119	0.45

5) *Sentence similarity using Fast-text:*
   - Run 01: Here, we have used the Fast-Text-based model. After training all the prior cases using the FastText model, we tested the prior cases. The result is presented in table VIII and X.

TABLE VIII
SCORE OF THE FASTTEXT BASED MODEL

MAP	Precision	Recall
0.275	0.115	0.565

   - Run 02: In this run, we removed stop words, using corpus and converted all the text context into a lower character form, and tested 2000 current cases. After that, we achieved quite a better result than the previous tests. The result is presented below

TABLE IX
SCORE OF FASTTEXT AFTER TUNING THE MODEL

MAP	Precision	Recall
0.019	0.133	0.589

   - Run 03: To improve the result further, we had to tune the model. We tuned the model by changing the epoch size. Previously we tested with 2 epochs and afterward, we tuned the model and used 5 epochs. Our result is presented below.

TABLE X
SCORE OF FASTTEXT AFTER TUNING THE MODEL

MAP	Precision	Recall
0.019	0.133	0.603

We sort the result based on the Recall value in which our system is set to the Number 1 position on the gold standard result (Table XI).

V. COMPARING OUR SYSTEM WITH SIMILAR SYSTEMS

We have finally selected the sorted N_similarity model after thoroughly analyzing each of the models. This model has been selected because it has a high MAP[12] and Recall value. The outcome of applying N_similarity was not great. But our system emerges at the first position when the similarity list is sorted. since the list was initially unsorted, the list is then sorted in ascending order according to similarity. The case with the most similarity score then emerges at the top positions. Overall, we achieved a satisfactory performance.

TABLE XI
SORTED N_SIMILARITY RANKING

position	Team	MAP	Precision	Recall	Method Summary
1	AustIRTeam	0.389	0.2009	0.976	sorted n_similarity
2	flt_ielab_idf	0.390	0.236	0.781	IDF, Citation Context
3	flt_ielab_para	0.364	0.221	0.749	Citation context
4	HLJIT2017_IRLeD_Task2_1	0.329	0.218	0.681	Dirichlet Prior Smoothing
5	SSN_NLP_2	0.268	0.178	0.669	TF-IDF(nouns+verbs)
6	SSN_NLP_1	0.263	0.180	0.681	TF-IDF(nouns)
7	HLJIT2017_IRLeD_Task2_3	0.248	0.167	0.671	lucene
8	rightstepspune_1_task2	0.202	0.135	0.564	RegEx, LDA, Doc2Vec
9	HLJIT2017_IRLeD_Task2_2	0.178	0.129	0.595	BM25
10	UB_Botswana_Legal_Task2_R3	0.167	0.123	0.559	DPH-DFR ,BoI model
11	UB_Botswana_Legal_Task2_R1	0.149	0.112	0.546	DPH-DFR
12	UB_Botswana_Legal_Task2_R2	0.108	0.079	0.43	DPH-DFR, Sequential Dependence
13	SSN_NLP_3	0.101	0.076	0.435	Word2Vec(nouns+verbs)
14	UBIRLeD_2	0.098	0.069	0.380	LDA
15	UBIRLeD_3	0.090	0.062	0.373	LDA
16	UBIRLeD_1	0.072	0.049	0.299	BM25
17	bphcTASK2IRLeD	0.071	0.060	0.280	POS tags, TF, IDF
18	AMRITA_CEN_NLP_RBG1_1	0.006	0.003	0.058	Doc2Vec
19	christfire_2017_3	0.005	0.003	0.033	LSA(nouns only)
20	christfire_2017_2 7	0.003	0.002	0.044	LSA
21	christfire_2017_1	0.002	0.003	0.016	LDA

VI. RESULT

From the table above, we can see that the recall parameter of the second and third team was good by using Citation context and the performance of the other method were good which used Dirichlet Prior Smoothing,
In the table XI, the runs are categorized by their MAP scores, but it should be highlighted that the Recall is particularly significant in a legal test-document retrieval. We know that the percentage of appropriate or pertinent documents that are successfully retrieved is known as recall and this is very important for legal practitioners that need pertinent documents. If we sort the result based on Recall then the result was in the Number 1 position on the gold standard result. So we can say that Recall would be a suitable evaluation method.

VII. FUTURE WORK

There are still more scopes to improve this proposed system. If we could take more documents for initial labeling, then there is a possibility of finding more appropriate relevant documents. The keyword-based data dictionary (for TF-IDF) still required to be improved. If the data dictionary could be developed as self-imposed, we may get better precision and recall. In the future, we may use some other model like GPT2[13], XLNet[14], Glove[15], etc. to see if there can be any performance gain. Moreover, we may try a hybrid model where we can merge all the results and find out the effective one, and that will be the final result. If we could use more text documents for learning purposes, there is a possibility of increasing the rate of Mean Average Precision. Our main target was developing an alternate query system that will be more efficient than existing systems. Though our system has achieved better results in precision and particularly in Recall value than the current systems, this system can be further improved to get better results.

VIII. DISCUSSION AND CONCLUSION

After evaluating the performances of all the runs of all the models, we can say that the ”Sorted N_Similarity” is currently the best model for this type of document retrieval system. We also see that, when we train the models without pre-processing the dataset, we get poor results. But when we pre-process our dataset, then the result gets improved. The result further improves when we tune the models. We also noticed that, when we tested the model with less data, then the average result is much better. But when we take all the current cases, then the average result drops. The reason behind this may be

- Some content of the dataset is too large.
- Lack of case-related keywords in the cases.

That is why the average result was not on the higher side.

REFERENCES

[1] Arpan Mandal, Kripabandhu Ghosh, Arnab Bhattacharya, Arindam Pal, and Saptarshi Ghosh, ”Overview of the FIRE 2017 IRLeD Track: Information Retrieval from Legal Documents”, FIRE 2017.
[2] S. Robertson and H. Zaragoza, The probabilistic relevance framework: BM25 and beyond. Now Publishers Inc, 2009.
[3] F. Song and W. B. Croft, A general language model for information retrieval, in Proceedings of the eighth international conference on Information and knowledge management, pp. 316321, 1999.
[4] M. McCandless, E. Hatcher, and O. Gospodnetic, ”Lucene in action, 2nd edn. covers apache lucene 3.0,” 2010.
[5] J. Ramos et al., ”Using tf-idf to determine word relevance in document queries, in Proceedings of the first instructional conference on machine learning, vol. 242, pp. 2948, Citeseer, 2003.
[6] A. A. Kimia, G. Savova, A. Landschaft, and M. B. Harper, An introduction to natural language processing: how you can get more from those electronic notes you are generating, Pediatric emergency care, vol. 31, no. 7, pp. 536541, 2015.
[7] A. Gross and D. Murthy, ”Modeling virtual organizations with latent dirichlet allocation: A case for natural language processing, Neural networks, vol. 58, pp. 3849, 2014.
[8] S. Sarsa et al., ”Information retrieval with finnish case law embeddings,” 2019.
[9] R. Subhashini and V. J. S. Kumar, A framework for efficient information retrieval using nlp techniques, in International Conference on Advances in Communication, Network, and Computing, pp. 391393, Springer, 2011.
[10] E. Hanafy and H. AbdelGelil, Places clustering based on sentiment analysis: A survey,, vol. 3, no. 3, pp. 15, 2021.
[11] J. Dalton, R. Blanco, and P. Mika, Coreference aware web object retrieval, in Proceedings of the 20th ACM international conference on Information and knowledge management, pp. 211220, 2011.
[12] MAP https://en.wikipedia.org/wiki/Evaluation_measures _(information_retrieval)#Average_precision. Accessed: 2022-08-30.
[13] Kanwal Ameen and Midrar Ullah. Information literacy instruction: An overview of research and professional development in Pakistan. In: European Conference on Information Literacy. Springer. 2016, pp. 555562.
[14] Jimmy Lin, Rodrigo Nogueira, and Andrew Yates. Pretrained transformers for text ranking: Bert and beyond. In: Synthesis Lectures on Human Language Technologies 14.4 (2021), pp. 1325
[15] Sumeer Gul, Sabha Ali, and Aabid Hussain. Retrieval performance of Google, Yahoo and Bing for navigational queries in the field of life science and biomedicine. In: Data Technologies and Applications 54.2 (2020), pp. 133150.