A Comparison of Techniques for Retrieval of Legal Documents

Asrita Venkata Mandalam
2017A7PS1179P
BITS Pilani
Pilani, India
f20171179@pilani.bits-pilani.ac.in

***Abstract*—The Forum for Information Retrieval 2019 Artificial Intelligence for Legal Assistance (FIRE 2019 AILA) track, Task 1, aimed to identify relevant prior cases for a given situation (Precedent Retrieval). Looking for previous cases relevant to the one at hand is a time-consuming process. This task aspired to automate this process. The dataset was taken from the Indian Supreme Court judiciary. This paper aims to provide a comparison between three teams' methods of approach to the task. HLJIT2019-AILA, SSN_NLP and JU_SRM submitted their models for the same track and were ranked on the basis of the Mean Average Precision (MAP), Precision at 10 (P10), BPREF and recip_rank scores achieved.**

***Index Terms*—Information Retrieval, Artificial Intelligence, Retrieval of Precedents**

I. Introduction

While deciding the outcome of a given case, courts are obliged to follow the same path of decisions that any previous cases or ruling might have taken. It's important for them to do so as this makes sure that our legal system stays fair and just by treating every case the same way. Sometimes, cases don't have any previous examples to follow. The ruling of this case is the ruling of a new situation and will serve to guide any future cases that are similar to it. These cases are called precedents. When lawyers plan to take up a case, reviewing these documents form an integral part of the preparation process.

This task aims to automate this as it would be more cost and time efficient for lawyers and employers of lawyers. Retrieving relevant documents helps the user predict the potential outcome of taking their case to court. It can also prevent a user from taking up a case that has a low probability of a favorable ruling. If released for public usage, this system could also help those who are not financially well-off by providing them with a medium to test whether filing a case and recruiting a lawyer is worth the money that they will spend on it.

In an attempt to review which model is better and the reasons for the same, this paper reviews the attempts of three teams. Each of these three teams submitted three runs each and received a ranking based on the MAP score their model achieved. Team SSN_NLP used Word2Vec, combined Word2Vec and GloVe, and finally, used a TF-IDF vectorizer for their three models. Team JU_SRM approached this task with a Sent2Vec embedding-based model, a FastText model and their last model was an ensemble of the two. HLJIT2019-AILA used a BM25 model for their first run, a modified version of the same for their second run and a Word2Vec based model for their last run.

II. Related Work

Essentially, this task is based information retrieval and is dependent on context-specific knowledge. Sugathadasa, et al [1] decided to combine text preprocessing with a postings list for their document retrieval system. Using similarity measures, neural networks and the original document, three document vector models were made. After that, they used them to train a neural network and tested their results. Overall, they proved that the neural networks approach was the best and achieved a higher accuracy than the rest. Srikanth, et al [2] went with biterms. They noticed that bigrams took into account the order in which words were present. However, the ordering is not always as important as it is expected to be and it becomes necessary to consider words in the same sentence or context as a pair (biterms). Gomez-Perez et al [3] created an ontology based legal document retrieval system. As legal data retrieval is very context specific, a perceptron algorithm was developed that could train a classifier for each pair of labels (Mencia et al)[4]. Perceptrons are classifiers that update the weights of nodes as soon as data arrives. They don't need to wait till all the data arrives, unlike Support -Vector Machines (SVMs).They did not want to lose out on accuracy but at the same time, wanted to deal with very large datasets. As perceptrons work in an incremental fashion effectively, they decided to use this algorithm instead of an SVM based algorithm. They managed to solve the storage space issue of their model as well by using dual multilabel pairwise perceptrons.

A similar task was released by FIRE organizers in 2017. Team christfire_2017 submitted a run which included Latent Semantic Analysis (LSA). After preprocessing their data to remove context-specific stopwords, they applied LSA and checked the similarity scores of their results [5]. They also submitted a run that ran through similar steps, except that they replaced LSA with Latent Dirichlet Allocation (LDA). Team rightstepspune_1_task2 used a weighted combination of regular expression based retrieval, document vector based and LDA for their run. The regular expressions part of their solution looked for patterns in the query. These patterns were

used by their model to find the most relevant documents. For the document vector part, they mapped the word to its context. The similarity score for the LDA model was generated by looking at the ratio of specific words to the total words. These specific words were chosen based on the topic.

III. Dataset

The AILA task creators included 2914 precedents. All of the documents are previous cases ruled by the Indian courts of Law. The participating teams were given 50 queries to test their model. For 10 queries, the task-setters provided a few relevant case documents. The remaining 40 queries required the participants to retrieve relevant documents [6].

IV. Methodology

A. Team SSN_NLP

Their first method used Word2Vec. Word2Vec is a two-layer neural network that can produce word embeddings. They made a model with a dimension of 300. They vectorized the whole document and found the cosine similarity between the query vector and the document vector. As the aim of this task was to produce the most relevant results (as opposed to balancing it with a certain time constraint), dimensions of 300 and above have been proven to produce the best accuracy (as used extensively by Mikolov et al) [7].

Their second method combined Word2Vec and GloVe embeddings. They vectorized all of the queries and documents using the two embeddings and then concatenated them. They found the cosine similarity between the query and document vector in this method as well. Their third method used a Term Frequency - Inverse Document Frequency (TF-IDF) vectorizer. Their vocabulary corpus consisted of words from the queries and cases. Finally, they found the cosine similarity between the query and document vector[8].

B. Team JU_SRM

This team submitted three models for task 1. Their first model used a pretrained word embedding called Sent2Vec. It predicts target words from source word sequences. Their second method used another word embedding, FastText, which predicts target words from character sequences. For both of these models, they extracted sentences with a maximum size of 20 tokens. They were working on prepossessed summarized documents instead of actual ones and in that setting, they got the value of 20 by looking at importance and relevance of selected tokens with respect to the other cases. Then, they used the respective word embeddings to encode them. After taking the average of all these vectors to get the query or case vector, they calculated the cosine-similarity score and ranked them. For their final model, they calculated the relative weights of the first two models using the given formula where $model_j$ and $model_i$ refer to specific models and *n* refers to the total number of models [9].

$$\frac{BPREF(model_j)}{\sum_{i=1}^{n} BPREF(model_i)}$$

This led to a weighted voting ensemble model which allowed multiple models to contribute their results according to their weights.

C. Team HLJIT2019- AILA

This team submitted three models as well. Their first one used the BM25 model to retrieve data. Using the Porter Stemmer, they stemmed their queries and documents. They removed stop words, punctuation and numbers. Next, they ranked the words in the query according to their inverse document frequency (IDF). Finally, they chose the top 50% to represent this query. Their retrieval model is defined in the equation below where w_i is the word in *q*, *avdl* is the average length of the document, k_1 and *b* are the parameters of BM25 [10].

$$rel(q,d) = BM25(q,d) =$$
$$\sum_{w_i \in q} log(\frac{(tf(w_i,d) \cdot idf(w_i)}{tf(w_i,d) + k_1 \cdot (1 - b + b \cdot \frac{LEN(d)}{avdl})}$$

For their second submission, they decided to go with a modified version of their first submission's model. All of the variables and parameters held the same value except for q". This model did not use the IDF approach so q" contained the query's value before the IDF ranking. Their relevance function's equation is defined in the equation below where q' is the same as the previous equation and q" is the without IDF filtering [10].

$$rel(q,d) = BM25(q',d) + BM25(q'',d)$$

Their third method used the Word2Vec equation specified below. t_i is the word vector of the i^{th} term and *n* is the top 50% terms [10].

$$V(x) = \frac{1}{n} \sum_{(i=1)}^{n} t_i$$

First, they ordered the words according to their TF-IDF score. They chose the top 50% of words using the above formula and represented the query and document vectors using the above formula. Finally, they found the similarity between the query and document by using Euclidian Distance.

V. Results and Error Analysis

A. Team SSN_NLP

Their Word2Vec model reached a MAP score of 0.0405 and was the highest MAP score out of the three runs that they submitted. Word2Vec works well as it requires less memory (less preprocessing) and it considers the context of words. However, the main disadvantage with this model is that words

and vectors have a one-to-one relationship. It does not account for the possibility that a word might have multiple meanings (polysemy). For example, it can't tell the difference between dusting a donut with powdered sugar and dusting a cupboard of cobwebs. Another disadvantage is that it can't handle out of vocabulary (OOV) words. It uses a random vector instead.

Their second method reached a MAP score of 0.0026. One might think that merging multiple sources of embeddings would do better than a single one. However, this is not always the case. Concatenation reuslts in a matrix with multiple redundant features. By using a projection from a certain number of m dimensions that reduces it to m dimensions, this redundancy is removed (Robyn et al) [11]. As both of these embeddings don't consider OOV words, it can lower the overall accuracy of this model.

Out of the three methods that SSN_NLP submitted, this performed the worst with a MAP score of 0.0025. This model can filter out unnecessary noise such as common and irrelevant words. However, one of the main issues with using TF-IDF alone is that it doesn't consider the semantic similarities between words in the same context. Each word and its count is considered to occur independently of any other word.

B. Team JU_SRM

Their Sent2Vec model received a MAP score of 0.0478, the highest of the three that JU_SRM submitted, and a rank of 14. This model boasts of a computational complexity of O(1) vector operations per word processed (Pagliardini et al) [12]. Another advantage is that the model learns to down-weight frequent tokens by itself.

The FastText model received a MAP score of 0.0228. Its memory consumption is not high however it is slower than GloVe and Word2Vec [13].

Their ensemble model reached a MAP score of 0.0181. They decided to do weighted average of FastText and Sent2Vec for feature aggregation from different modality of data. The results were poor as there were irrelevant words and sentences in the case documents which acted as outliers in the context domain. Taking into account Named Entity Recognition (NERs), removing irrelevant topics and words would have improved their model.

C. Team HLJIT2019-AILA

Their improved BM25 model received a MAP score of 0.1492 and was the highest among all of the submissions for AILA task 1.

The first method's BM25 model received a score of 0.1335 and was the third highest in the same task. As probabilistic relevance models have been tested and tried to provide good results (Zhai, 2017) [14], it is no surprise that this BM25 model performs the best. The top few runs of this task applied BM25 and TF-IDF as well. An attempt to solve a similar task by Competitionon Legal Information Extraction/Entailment

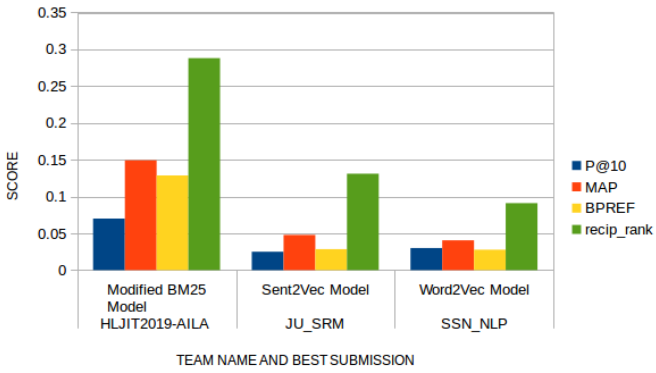

Fig. 1. Summary of Results

(COLIEE-2019) reinforced the same conclusion (Gain et al) [15].

Their Word2Vec model got a MAP score of 0.0220. The main reason for such a low score was the usage of a general corpus for word vector pre-training. As it wasn't context specific, it wouldn't have been able to retrieve many relevant documents (their BPREF was 0.0066). Also, there might have been multiple OOV words and Word2Vec does not handle it well.

VI. Conclusion and Future Work

From the results of each of the teams' models and from the other teams that participated in this FIRE 2019 AILA track, we can conclude that the BM25 model, with taking into consideration the Inverse Document Frequency (IDF) scores, performs the best out of all the submitted systems. Team HLJIT2019-AILA performed the best with their modified BM25 model. Their MAP score was 0.1492 and it can definitely be improved upon. While using pre-existing word embeddings is a good direction, the lack of inclusion of terminologies used in legal documents, the inability to deal with OOV words and insufficient preprocessing prevent the other teams' models from reaching a higher MAP score.

Taking into account domain-specific terminology forms an integral part of a legal information retrieval system. Legal jargon is vital as these words may not be used as frequently and with the same implication outside of these documents. The importance (weights) of these words should be increased or decreased accordingly. While TF-IDF models might take care of the frequency problem, word embeddings won't take the latter into consideration. Thus, some manual annotation or context specificity might be required if a model would prefer to use word embeddings.

Legal documents have a fixed structure and precedents tend to have long, complex prose. This retrieval system can be used by those who do not know complex terminology and are not in the field of law and legal proceedings. These people might know the concept of what they would like to retrieve in layman's terms. Mapping their queries to the corresponding

legal jargon is an important and useful feature that can be added to this system. This can be implemented in the form of a dictionary of terminology synonyms. To make the system faster, common phrases that are mapped together can be stored separately. This is just a layer that can be added to any legal data specific retrieval model.

It has been seen that apart from TF-IDF based models, recurrent neural networks (RNN) such as Long Short Term Memory (LSTM) have been used in the past for information retrieval, although, not specifically for retrieving legal documents. A paper by Luukkonen et al [16] shows the use of LSTMs to predict what users will be searching for based on their previous searches. Deep learning models do take a longer time to retrieve results but considering that the aim is to reduce the amount of time and the number of unnecessary precedants the user will have to go through, it seems appropriate to apply this here. Also, with the help of the LSTM model mentioned previously and taking into consideration legal terms and their usage, queries related to the first one will draw more relevant documents. Overall, the time spent on the research of a case would reduce by a noticeable margin and would also produce the required results.

Another RNN based retrieval system has been described by Pang et al [17]. DeepRank finds relevant context based on the query vector. They apply a two-dimensional gated recurrent unit (2D-GRU) to measure exactly how relevant it is. Finally, they use an RNN to aggregate their previous result. While this has been shown to outperform other models, it has been used to test general queries. In this case, the context is already known. We can skip over the detection phase and replace it with a preprocessing step that can tag words with an appropriate measure of weights. The 2D-GRU can work on these weights to produce results for the next step. We can also try to classify the precedents into subtopics by using clustering. That way, if a query clearly belongs to a specific cluster, the model can reduce its work to that specific range only. This would increase the BPREF score and reduce the number of irrelevant documents that show up in the top k documents, where k is the required number of results to be retrieved.

Lawyers spend a lot of time reading through previous cases to come to a conclusion whether it would be useful or not. This is where text summarization plays an important role and can add to the value of the retrieval system. Information can be extracted from the list of relevant results and be displayed below each of the retrieved document. To choose which data can be taken, the model should implement parts of speech (POS) tagging. It should include details about the case such as background information and facts. The outcome is not as relevant as the previously mentioned elements while displaying a summary. The TextRank algorithm, which is an unsupervised technique, can be used for this purpose along with a few modifications. It uses word embeddings and with the appropriate changes in the weights of terms, it can be used efficiently.

VII. Acknowledgment

I would like to thank my instructor for the assignment, Dr. Abhishek, for providing this learning opportunity. Without his constant support, the completion of this report would not have been possible. His guidance has been invaluable as it helped me understand different topics in the course and has helped me apply them to approach problems and find new and better solutions.

References

[1] Sugathadasa, K., et al.: Legal document retrieval using document vector embeddings and deep learning. In Arai, K., Kapoor, S., Bhatia, R. (eds.) SAI 2018. AISC, vol. 857, pp. 160–175. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-01177-2_12.

[2] Srikanth, M. And Srihari, R. 2002. Biterm language models for document retrieval. In: SIGIR 2002, pp. 425-426.

[3] A. Gomez-Perez, F. O. Rodriguez, B. Villazón-Terrazas, "Ontology-Based Legal Information Retrieval to Improve the Information Access in E-Government", 3rd European Semantic Web Conference (ESWC 2006), 11–14 June, 2006.

[4] E. Loza Mencía, J. Fürnkranz, W. Daelemans, B. Goethals, K. Morik, "Efficient pairwise multilabel classification for large-scale problems in the legal domain" in Lecture Notes in Artificial Intelligence 5212, Berlin, Germany:Springer, pp. 50-65, 2008.

[5] Mandal A, Ghosh K, Bhattacharya A, et al. Overview of the FIRE 2017 IRLeD Track: Information Retrieval from Legal Documents[C]//FIRE (Working Notes). 2017: 63-68.

[6] Ravina More, Jay Patil, Abhishek Palaskar and Aditi Pawde, 2019. Removing Named Entities to Find Precedent Legal Cases. In FIRE 2019.

[7] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient Estimation of WordRepresentations in Vector Space. In ICLR Work-shop Papers.

[8] Kayalvizhi, S., Thenmozhi, D., Aravindan, C.: Legal assistance using word embed-dings. In: Proceedings of FIRE 2019 - Forum for Information Retrieval Evaluation(December 2019).

[9] Mandal, S., Das, S.D.: Unsupervised identification of relevant cases and statutes using word embeddings. In: Proceedings of FIRE 2019 - Forum for Information Retrieval Evaluation (December 2019).

[10] Zhao, Z., Ning, H., Huang, C., Kong, L., Han, Y., Han, Z.: Fire2019@aila: Legal information retrieval using improved bm25. In: Proceedings of FIRE 2019 - Forum for Information Retrieval Evaluation (December 2019)

[11] Robyn Speer and Joanna Lowry-Duda. 2017. Con-ceptnet at semeval-2017 task 2:Extending word embeddings with multilingual relational knowledge. In Proceedings of the 11th In-ternational Workshop on Semantic Evaluation(SemEval-2017). Association for Computational Linguistics, Vancouver, Canada, pages 85-89.

[12] Matteo Pagliardini, Prakhar Gupta, and Martin Jaggi.2017.Unsupervised learning of sentence em-beddings using compositional n-gram features.arXiv:1703.02507.

[13] E. Alvarez and H. Bast, "A review of word embedding and document similarity algorithms applied to academic text," 2017.

[14] Zhai, C. A Brief Review of Information Retrieval Models, Technical report, Dept. of Computer Science, UIUC, 2007.

[15] B Gain, D Bandyopadhyay, T Saikh, A Ekbal: IITP in COLIEE@ ICAIL 2019: Legal Information Retrieval using BM25 and BERT, 2019.

[16] Luukkonen, P., Koskela, M., and Floreen, P. (2016). LSTM-Based Predictions for Proactive Information Retrieval. In ACM SIGIR Workshop on Neural Information Retrieval (Neu-IR). arXiv:1606.06137.

[17] Liang Pang, Yanyan Lan, Jiafeng Guo, Jun Xu, Jing-fang Xu, and Xueqi Cheng. 2017.Deeprank: A new deep architecture for relevance ranking in information retrieval. In Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, pages 257–266. ACM.