CUSAT_NLP@AILA-FIRE2019: Similarity in Legal Texts using Document Level Embeddings

Sara Renjit[1] and Sumam Mary Idicula[2]

[1] Research Scholar, Dept. of Computer Science, CUSAT, Kochi-682022
sararenjit.g@gmail.com
[2] Professor, Dept. of Computer Science, CUSAT, Kochi-682022
sumam@cusat.ac.in

Abstract. Text retrieval has taken its role in almost all domains of knowledge understanding. It has applications in the legal field where there is an extensive collection of structured and unstructured texts. Artificial Intelligence is now applied in this area to understand and retrieve legal documents. This paper explains a working model developed for the track Artificial Intelligence for Legal Assistance in Forum for Information Retrieval Evaluation, 2019 (AILA-FIRE2019). We have used an embedding model approach to represent these legal texts in a semantic vector space. The similarity between these document embeddings is found using an existing method of cosine similarity. The corpus used for building embedding models is the dataset provided in AILA-FIRE2019.

Keywords: Legal texts · Similarity · Precedents · Statutes · Embeddings.

1 Introduction

Nowadays, the amount of legal content available from online sources is high. The rapid change in handling legal documents and related pieces of information in electronic form has increased the need for using artificial intelligence in this domain. Interfaces that provide a semantic understanding of texts would be useful for non-specialists as well as legal practitioners in understanding statutes and related prior cases. A layman gets the basic knowledge about legal proceedings by looking through the most relevant precedent cases and statutes retrieved for each query situation.

The rest of this paper is organized as follows: Section 2 presents related works in the area of document retrieval. Task description and dataset details are provided in Section 3. Section 4 explains the methodology used. Section 5 relates to experimental details and evaluation results. Finally, Section 6 concludes the work with some future improvements.

2 Related Works

Text retrieval also called document retrieval, matches some query against a set of text records either structured or unstructured. Methods to retrieve texts is more or less based on some similarity measures. Various clustering methods were also employed. Graph-based approaches were also used to find document similarity using modified shortest path graph kernals[5]. A dual embedding space model is proposed in [4], where both input and output word vector embeddings are used for word similarity estimation in document ranking. Text retrieval in the legal domain has been an emerging research topic nowadays, and few works are studied. A legal document retrieval system was developed for the general public in China using normalized Google Distance to find the semantic relatedness between layman terms and legal terms[2].

3 Task Description & Dataset Details

AILA Track consists of two tasks, namely precedent retrieval (Task 1) and statute retrieval (Task 2). Task 1 aims at finding relevant previous case documents/ precedents for a situation mentioned in the query document. Task 2 retrieves the most similar statutes for a scenario given in the query.

The table below shows the statistics of the dataset for both evaluations. It is a collection of Indian legal documents, i.e., statutes in India and prior cases decided by Indian courts of Law.[1]

Table 1. Dataset statistics

#Query	# Precedents	#Statutes
50	2914	197

4 Proposed Method

We have used document embedding model with cosine similarity to understand the similarity between documents. Paragraph vector [3], which is an unsupervised algorithm that learns feature representations from sentences and texts, is used as document embedding model in this context.

Paragraph vectors are inspired from architecture for learning word vector representations, in which document or paragraph id is assigned to each paragraph. Each word in a paragraph is represented by a column vector in a matrix, say W. Concatenation or sum of these vectors are features for predicting next word in the sentence. For sequence of words w_1,w_2,...,w_T word vector model maximizes the average log probability

$$\frac{1}{T}\sum_{t=k}^{T-k}\log p(w_t|w_{t-k},...,w_{t+k}) \tag{1}$$

Prediction is done using a softmax classifier.

$$p(w_t|w_{t-k}, ..., w_{t+k}) = \frac{e^{y_{w_t}}}{\sum_i e^{y_i}} \tag{2}$$

where, y_i can be computed as unnormalized log probability for each output word i using

$$y = b + Uh(w_{t\text{-}k}, ..., w_{t+k}; W) \tag{3}$$

where U, b are the softmax parameters. h is constructed by a concatenation or average of word vectors and paragraph vectors extracted from W and D respectively.

The paragraph vectors and word vectors are averaged or concatenated to predict the next word in a context. Paragraph vectors and word vectors are trained using stochastic gradient descent, and then paragraph vector for new paragraphs can be inferred. The two stages in paragraph vector algorithm are: 1) Training word vectors W, softmax weights U, b and paragraph vectors D on already seen paragraphs. 2) Inference step derives paragraph vectors for new paragraphs by adding more columns in D and gradient descending on D keeping W, U, b fixed.

There are two variations of this model: PV-DM (Distributed Memory Model of Paragraph Vectors) which considers word order with the concatenation of word vectors and paragraph vector. The missing context information is in the paragraph vector, and it memorizes the semantic content of the paragraph. PV-DBOW (Distributed Bag of Words model of Paragraph Vector) neglects word ordering by taking random samples of words to predict another word. Since word ordering is essential to understand the semantics of a text, a distributed memory model is preferred while training the paragraph vector model. Vector representation for documents are inferred from this model and are then compared using Cosine similarity.

Cosine similarity [8] measures the angle between two vectors in multidimensional space. Cosine of two vectors is : $A.B = |A||B|cos\theta$ and cosine similarity, given two vectors A and B, is defined as:

$$similarity = cos\theta = \frac{A.B}{|A||B|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}} \tag{4}$$

Using this similarity metric, we can understand the orientation of documents as its based on angle measurement. Documents with similar orientations can be considered as similar and if the document vectors are created in such a way that the semantics of element words are taken care of, then these orientations can also be semantically similar.

5 Experiments & Results

A workflow of the proposed system is presented in Figure 2. The collection of precedent documents, statutes and query cases are provided in form of text files

as input. The first step is tokenization of each document to a collection of tokens or words and then each collection is labelled with a tag and number to uniquely identify documents. This represents paragraph id in the paragraph vector model discussed above.

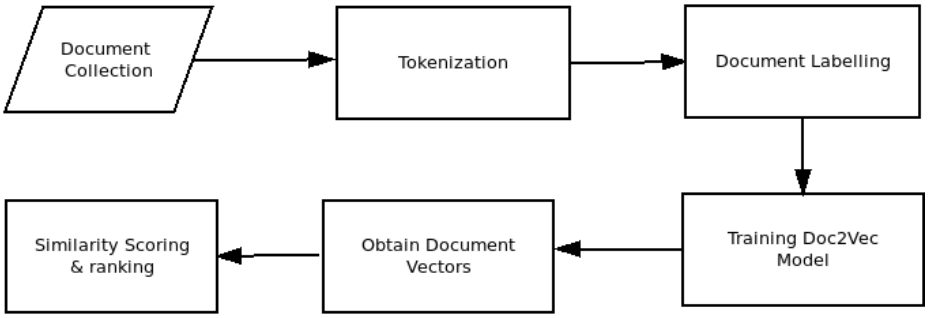

Fig. 1. Workflow of the proposed system

In the next phase, doc2vec model from gensim library is used for training the labeled precedents, query, and statutes. Doc2vec model is trained using the precedents and query documents collection with five epochs and embedding dimension set as 100. This model is then used to obtain the vector representation of each document for precedents retrieval task. Statutes collection is also added to the existing doc2vec corpus for retrieving vector representations for task 2. In the final stage, vector representations obtained for each case document, precedents and statutes are compared using cosine similarity.

For Task 1, cosine similarity between present case situation and precedents are calculated, sorted, and ranked. Similarly, for Task 2, cosine similarity between each present case situation and statutes are found. The similarity scores are then sorted and ranked. The final result gives a list of ranked documents for each case scenario presented in the query document.

The proposed system uses 50 cases as query documents out of which ten queries were used for pre-evaluation and 40 in the final evaluation. TREC Evaluation [6] is used to evaluate the system performance. Various evaluation measures used in this evaluation are:

- Precision@10: The precision (percent of retrieved docs that are relevant) after ten documents have been retrieved, and these values are averaged over all queries.
- Mean Average Precision: It is the average precision across multiple queries /rankings, and average precision is the average of all P@K (say 10).
- Binary Preference: It computes a preference relation of whether judged relevant documents are retrieved ahead of judged irrelevant documents.
- Reciprocal Rank : Reciprocal rank of top relevant document. [7].

The table below shows the results of the proposed system on both evaluations.

Table 2. TREC Evaluation results for both tasks

Task name	#Queries	P@10	MAP	BPREF	RR
Precedents retrieval	40	0.030	0.0481	0.0412	0.166
Statutes retrieval	40	0.055	0.0967	0.0377	0.199

6 Conclusion

This paper presented a paragraph vector and cosine similarity based document retrieval approach for legal documents. Legal document retrieval involves understanding the semantics of legal ontology. Efficient sentence modeling approaches that can semantically model legal documents can further improve the performance of this system. Semantic understanding needs to be further improved to distinguish and understand the similarity between legal terms and layman terms.

References

1. Bhattacharya, P., Ghosh, K., Ghosh, S., Pal, A., Mehta, P., Bhattacharya, A., Majumder, P.: Overview of the FIRE 2019 AILA track: Artificial Intelligence for Legal Assistance. In: Proceedings of FIRE 2019 - Forum for Information Retrieval Evaluation (December 2019)
2. Chen, Y.L., Liu, Y.H., Ho, W.L.: A text mining approach to assist the general public in the retrieval of legal documents. Journal of the American Society for Information Science and Technology **64**(2), 280–290 (2013). https://doi.org/10.1002/asi.22767, https://onlinelibrary.wiley.com/doi/abs/10.1002/asi.22767
3. Le, Q., Mikolov, T.: Distributed representations of sentences and documents. In: International conference on machine learning. pp. 1188–1196 (2014)
4. Nalisnick, E., Mitra, B., Craswell, N., Caruana, R.: Improving document ranking with dual word embeddings. In: Proceedings of the 25th International Conference Companion on World Wide Web. pp. 83–84. WWW '16 Companion, International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland (2016). https://doi.org/10.1145/2872518.2889361, https://doi.org/10.1145/2872518.2889361
5. Nikolentzos, G., Meladianos, P., Rousseau, F., Stavrakas, Y., Vazirgiannis, M.: Shortest-path graph kernels for document similarity. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing. pp. 1890–1900. Association for Computational Linguistics, Copenhagen, Denmark (Sep 2017). https://doi.org/10.18653/v1/D17-1202, https://www.aclweb.org/anthology/D17-1202
6. NIST: Evaluation scripts for text retrieval conference, http://trec.nist.gov/trec_eval/

7. Wikipedia contributors: Mean reciprocal rank — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Mean_reciprocal_rank&oldid=872349108 (2018), [Online; accessed 3-September-2019]
8. Wikipedia contributors: Cosine similarity — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Cosine_similarity&oldid=910391235 (2019), [Online; accessed 3-September-2019]