Precedent Case Retrieval using Wordnet and Deep Recurrent Neural Networks

Conference Paper · December 2019

DOI: 10.5121/csit.2019.91608

3 authors, including:

Sai Vishwas Padigi
PES Institute of Technology
1 PUBLICATION **0** CITATIONS
SEE PROFILE

Natarajan Subramanyam
PES Institute of Technology
77 PUBLICATIONS **187** CITATIONS
SEE PROFILE

PRECEDENT CASE RETRIEVAL USING WORDNET AND DEEP RECURRENT NEURAL NETWORKS

Sai Vishwas Padigi, Mohit Mayank and S. Natarajan

Department of Computer Science and Engineering, PES University, Bengaluru, India

ABSTRACT

The slowness of legal proceedings in the common law legal system is a widely known fact. Any tool which could help reduce the time taken for the resolution of a case is invaluable. Common legal systems place a great importance on precedents and retrieving the correct set of precedents is considerably time consuming. Hence, for any case whose proceedings are in progress, if there are suitable prior cases, then the court has to follow the same interpretations that were passed in the prior cases. This is to ensure that similar situations receive similar treatment, thus maintaining uniformity amongst the legal proceedings across all courts at all times. Hence, precedent cases are treated as important as any other written law (a statute) in this legal system. In this paper, we propose two new approaches to solve this information retrieval problem wherein the system accepts the current case document as the query and returns the relevant precedent cases as the result. The first approach is to calculate the document similarity using Wordnet, which is a lexical database that could be leveraged to quantify the semantic relatedness between two documents, using a semantic network. The second approach is the use of a Siamese Manhattan Long Short Term Memory network, which is a supervised model trained to understand the underlying similarity between two documents.

KEYWORDS

Information retrieval, Text similarity, Deep learning, Legal documents, Wordnet, Siamese Manhattan LSTM

1. INTRODUCTION

In India, the average time required for the legal proceedings associated with any case to be resolved is around 2 years [1]. According to data from a 2000 survey, there were 3.1 million cases pending in High Courts and 20 million in subordinate courts at the time of the survey [1]. As of 2019, the pendency is at its highest with 33 million cases pending across all Indian courts (28.4 million pending cases in the subordinate courts, 4.3 million pending cases in the High Courts and 57,987 pending cases in the Supreme court). 60% of all the pending cases are more than 2 years old and 40% are more than 5 years old [2]. In one extreme example of judicial slowness, a case had taken an excruciating time of 47 years to be resolved [1]. Hence, any tool which could reduce the time taken for the disposal of a case is invaluable.

Precedent case retrieval is one particular task that consumes a significant amount of time. Precedent or prior cases associated with a current case refer to cases that the court has come across in the past and are similar in nature to the current case [3]. The Common Law System [4], which is one of the most followed legal systems in the world bestows significant importance to precedent cases. The Common Law System's *stare decisis* principle states that similar facts or

Natarajan Meghanathan et al. (Eds) : DMAP, CSITA, ARIN, ISPR, AISO - 2019
pp. 85-98, 2019. © CS & IT-CSCP 2019 DOI: 10.5121/csit.2019.91608

similar situations need to be treated similarly [5]. Hence, for any case whose proceedings are in progress, if there are suitable precedent cases, then the court has to follow the same interpretations that were passed in the prior cases [3]. This is to ensure that similar situations receive similar treatment, thus maintaining uniformity amongst the legal proceedings across all courts at all times. Hence, precedent cases are given the same weight and treated as important as any other written law (a statute) in this legal system [5].

Due to the criticality associated with the precedent cases, it is highly essential for legal professionals to have access to the relevant precedent cases during the legal proceedings of any case. Typically, lawyers examine the judgments passed in the precedent cases relevant to the case they are dealing with in order to prepare for the arguments in advance [6]. On the other hand, the judges also refer to the precedent cases so as to align their judgments with those in the precedent cases. The judges are obliged to do this as they need to ensure that similar cases are dealt with in a similar and consistent manner.

Thus, in order to find the relevant precedent cases, the judges and lawyers need to search for them from a very extensive database of prior cases. In addition to the sheer number of prior cases to be scanned through, the huge size of each of these prior case documents makes it harder to find relevant prior cases manually [7]. Due to this reason, building efficient precedent case document retrieval mechanisms in the legal domain is a research issue [8]. Measuring the similarity between two court case documents is a challenge of its own due to the complex structure and lengthy nature of legal documents [5].

In this paper, we propose two new approaches to solve this information retrieval problem wherein the system accepts the current case document as the query and returns the relevant precedent cases. The first approach is to calculate the document similarity using Wordnet, which is a lexical database that could be used to quantify the semantic relatedness between the documents using a semantic network. The second approach is to train a Siamese Manhattan Long Short-Term Memory (MaLSTM), which is a supervised model to understand the underlying semantic similarity between two documents. This model can then be used to compute the semantic similarity between two legal court case documents.

2. DATASET

For this task, we have used the dataset made available by the Forum for Information Retrieval and Evaluation for their IRLeD track [3]. This dataset primarily consists of a set of 200 current court case documents, each of whose precedent cases are to be retrieved from a pool of 2000 precedent case candidate documents.

These court case documents capture the entire legal proceedings of a case in court. In addition to text-based information, these documents contain links to the precedent cases known as citations, similar to references in research articles [3]. However, for the purpose of this retrieval evaluation, all citations in the set of current case documents are masked with the [?CITATION?] marker. Out of the 2000 precedent case candidate documents, only 1000 of them are cited in the current case documents, whereas the remaining 1000 documents are not cited in any of the current case documents.

We use the given labeled dataset to form our training data. The dataset is conformed to the format of X_{left}, X_{right}, Y, where X_{left} = current case document, X_{right} = precedent case document, Y = 1 or 0 (1 if Y is actually a precedent of X, 0 otherwise). From the given dataset, we are able to create all the positive data points. We pick up to 4 random precedents from the pool for each current case documents. This forms the set of our negative data points. Thereafter, we split the data into training and validation sets with a ratio of 80 to 20.

3. LITERATURE SURVEY

There has been a significant amount of research effort that has gone into precedent case retrieval using text-based methods. The most common approach has been to formulate a query out of the current case document and to use this query to compute its similarity with each of the possible precedent case document candidates. The most similar precedent case candidate documents are predicted as the most relevant precedent cases for the corresponding current case document.

With respect to query formulation, there are a handful of works that have leveraged the part of speech (POS) tagging associated with each of the words [9][10]. Queries have been formulated consisting only of nouns or a combination of nouns and verbs [9]. Other approaches include considering only the legal terms as part of the query [6]. Summarizing the entire document and using this summary as a query [5] has been attempted but one drawback with such an approach is that the quality of the summary could adversely affect the accuracy of the precedent document retrieval system. Other interesting approaches to query formulation include the comparison of paragraphs, the statutes mentioned in the documents [11], the reason for citation i.e. the text surrounding a citation [5][12] and topic modeling using techniques such as LDA [11].

Post query formulation, the queries, and the precedent case candidate documents are converted to vectors before comparing them. The TF-IDF [13] vectorizer has been used in a lot of works [5][6][9] for this purpose. Of late, embeddings such as Word2Vec [14] and Doc2Vec [15] have gained prominence and there are several works leveraging the same [5][16][9][11]. The similarity between the two vectors has been computed using cosine similarity in most works [5][6][16][9]. Apart from these techniques, the Okapi BM25 ranking has also performed well in retrieving relevant documents [12].

While all the above-mentioned works are text-based approaches, there has been some progress on network-based similarity measures. In this approach, the citation network of legal documents is formed by a directed graph containing the different documents as the nodes and each directed edge signifying the cited document and the document containing the citation. Network-based similarity measures such as the bibliographic coupling and co-citation were proposed which take into consideration the number of out-citations and in-citations respectively between two documents [6]. Despite the network-based approaches having performed well, they cannot be readily used. This is because these legal documents form a very sparse citation graph whereas these methods require a well-connected citation graph to achieve a good performance [5].

The traditional TF-IDF based models have performed well in the past but are limited to understanding the context by the inherent term specificity. The ability to understand the semantic expressiveness of natural language took a huge leap forward with the invention of Word2Vec and Doc2Vec. In this paper, we propose two approaches to take a step in the same direction to retrieve relevant precedent cases based on semantic similarity. In our first approach, we propose precedent case retrieval leveraging Wordnet, which is capable of quantifying the semantic relatedness between words using a semantic network. In our second approach, we propose precedent case retrieval using a Siamese Manhattan LSTM network, which is a supervised model trained to understand the underlying semantic similarity between two texts. The Siamese Manhattan LSTM has performed exceptionally well in assessing the semantic similarity between two sentences and we aim to extend the model to assess the semantic similarity between two legal case documents.

Amongst all the approaches that have been published using the same dataset as that as we are using, Locke et al. [12] produced the best results with a mean average precision of 0.3902. We consider this as the baseline against which we compare our present work.

4. BACKGROUND

4.1. Wordnet

Wordnet [17] is a large lexical database comprising English words. The different parts of speech such as nouns, verbs, adjectives, and adverbs are clustered into cognitive synonyms, also known as synsets. Each of these synsets represents a distinct concept. Additionally, these synsets are interconnected by means of conceptual-semantic and lexical relations [17].

4.2. Siamese Manhattan LSTM

The Long Short-Term Memory model (LSTM) [18] was essentially built upon the Recurrent Neural Network (RNN) to overcome the primary limitation of the RNN to work with long sequences due to vanishing gradients. LSTMs have the ability to retain useful information while dropping unnecessary information. Hence, LSTMs and its variants such as the Gated Recurrent Units (GRU) [19] are capable of encoding the meaning of sentences into fixed-length vector representations. Hence, LSTMs could be leveraged as supervised learning models that take the sentence pairs and the underlying similarity between the sentence pairs as input and produce a mapping from a general space of variable length text sequences into an interpretable representation with a fixed dimensionality vector space.

The Siamese LSTM model comprises two LSTM networks, each of which processes one of the sentences in an input pair and the weights of these two LSTMs are tied or shared. Each LSTM takes as input a vector representation of the sentence and outputs a fixed dimensional hidden state representation encoding the semantic meaning of the sentence. These hidden states of the two LSTMs are compared against each other using the Manhattan distance to output a similarity score. Hence, the name Siamese Manhattan LSTM (MaLSTM) [20].

5. METHODOLOGY

5.1. Wordnet for Precedent Case Retrieval

In this approach, we aim to leverage Wordnet as a knowledge-based measure that could quantify the semantic relatedness between words using a semantic network.

5.1.1. Query Formulation

In general, each legal case document contains multiple citations to precedent cases. These precedent cases could be cited while discussing different legal issues or in different contexts. Hence, using the entire current case document as the query would lead to generalization and a loss of context with respect to the citations. Such a query would return precedent cases whose overall theme is similar to that of the current case but would not be very effective at retrieving precedent case documents relevant to each of the citations. A better approach would be to formulate individual queries for each of the citations and retrieve the relevant precedent cases using this set of queries.

Additionally, most legal documents discuss several legal issues based on the nature of the court case being described. However, each paragraph in the document usually discusses only one specific legal issue [5]. We leverage this statement in order to capture the Reason for Citation [5]. For each of the citations, the paragraph containing the citation forms the query, thus effectively capturing the context around each citation.

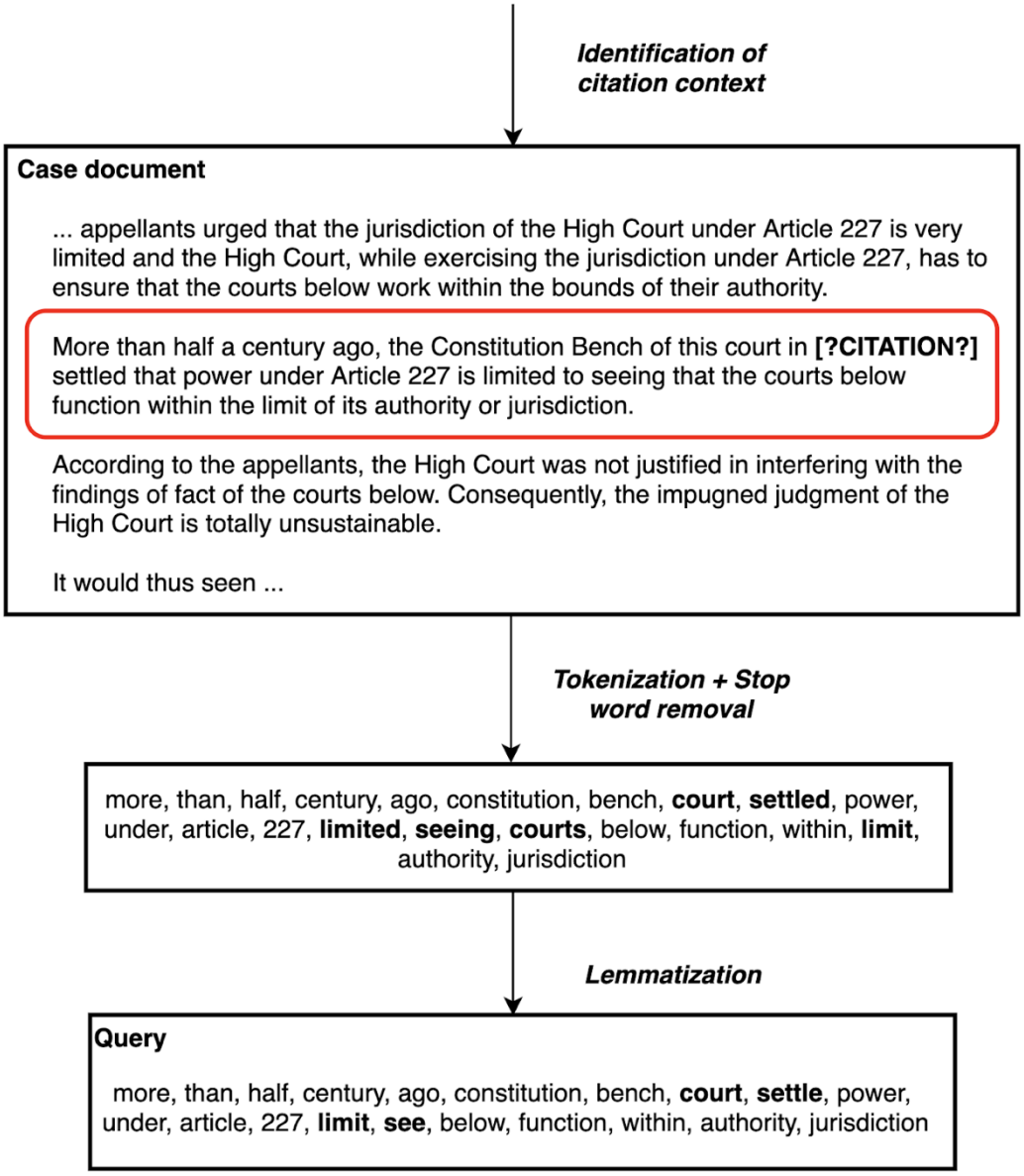

Figure 1. Steps involved in Query Formulation

Each citation in the current case documents is masked by the citation marker '[?CITATION?]'. Thus, each paragraph in the document containing this citation marker forms a query. Prior to the similarity comparison with the precedent cases, all stop words are removed from these queries post tokenization as shown in Figure 1. Additionally, we have also lemmatized the words in the queries using the Wordnet lemmatizer. We have used the Wordnet lemmatizer as it performed better than Porter stemmer [21] and Lancaster stemmer [22] on our dataset. Tokenization, stop word removal and Wordnet lemmatization have been performed on all the precedent case candidate documents as well.

5.1.2. Similarity Measurement using Wordnet

Post preprocessing, each of the formulated queries and each of the precedent case candidate documents was stored in the form of a list of tokens. Using Wordnet, each of these tokens was

converted into a synset, ie a cluster of cognitive synonyms based on the original word itself and its form of usage (i.e Part of Speech) in the original sentence. Post this step, we had each of the queries and each of the precedent case document candidates represented by individual lists of synsets.

The similarity between a query and a precedent case document was computed by measuring the similarity between the corresponding lists of synsets. For each synset in the query, we found the synset in the precedent case document with the largest similarity value. The sum of all the largest similarity values across all the synsets in the query list was normalized by dividing this sum by the total number of largest similarity values found. This score represents the similarity between the query and the precedent document.

The similarity between the two synsets was evaluated leveraging Wordnet's inbuilt similarity evaluation methods. The thesaurus based similarity measures included the Path distance similarity measure, Leacock Chodorow similarity measure [23] and the Wu-Palmer similarity measure [24] whereas the thesaurus and corpus-based similarity metrics, known as Information Content metrics included the Resnik similarity [25], Jiang-Conrath similarity [26], and Lin similarity measures [27].

Amongst the thesaurus based similarity measures, we observed the best results using the Leacock Chodorow similarity measure, which returns a score denoting how similar the two word senses are. This is computed based on the shortest path connecting the senses and the maximum depth of the taxonomy in which the senses occur. The similarity is formulated as

$$-log(p/2d)$$

where p is the shortest path length and d is the taxonomy depth.

Amongst the thesaurus and corpus based similarity measures, we observed the best performance using the Jiang Conrath similarity measure, which also returns a score denoting how similar the two word senses are. This is computed based on the Information Content (IC) of the Least Common Subsumer (most specific ancestor node) and that of the two input Synsets. The more abstract a concept, the lower its information content. The information content of a concept c is quantified as negative the log likelihood i.e.

$$IC = -log\ p(c)$$

Information Content provides for a quantitative characterization of the information, which provides a way to measure semantic similarity. The similarity is calculated using the equation:

$$1 / (IC(s1) + IC(s2) - 2 * IC(lcs))$$

where IC is the Information Content, s1 and s2 are the two synsets and lcs is the least common subsumer.

5.1.3. Ranking Precedent Documents in Order of Relevance

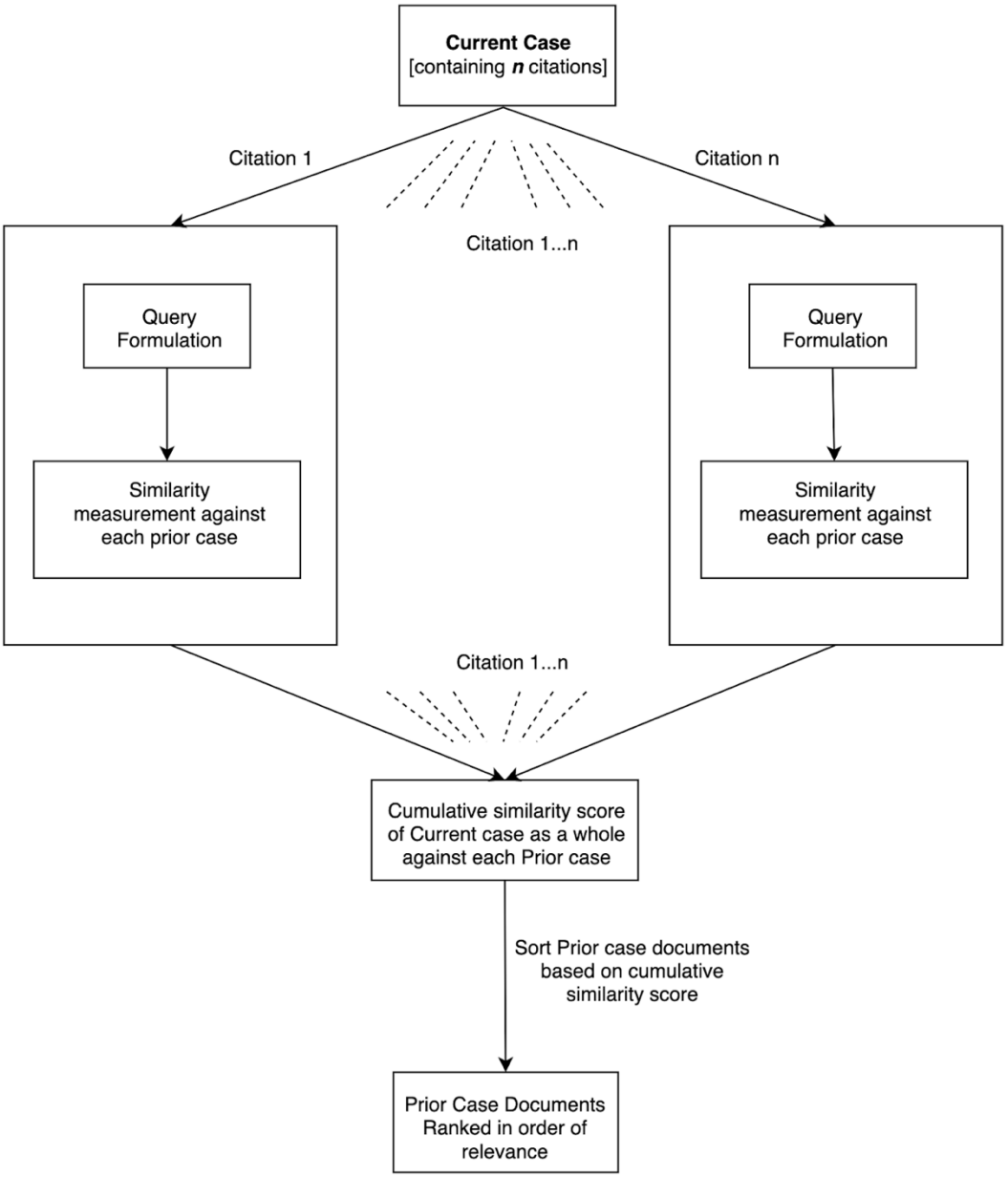

Figure 2. The flow diagram of Precedent case retrieval system using Wordnet

Using the above-mentioned technique, the similarity measure between each of the queries and the precedent case documents was computed. Since each current case document contained a number of citations, there were multiple queries generated from a single current case as shown in Figure 2. The cumulative list of precedent cases relevant to a current case was generated by choosing the maximum similarity score for each precedent case across all the queries. For each current case the 2000 precedent cases were sorted in descending order of their similarity score producing a ranked list of precedent cases retrieved.

5.2. Siamese Manhattan LSTM for Precedent Case Retrieval

For Information Retrieval purposes, it is suggested to have the weights of the two LSTMs untied as the query text may differ stylistically from the document it is being compared against. But in our approach, we have consciously implemented the Siamese architecture wherein the weights of the two LSTMs are tied or shared as we are comparing two legal case documents against each other to evaluate their semantic similarity.

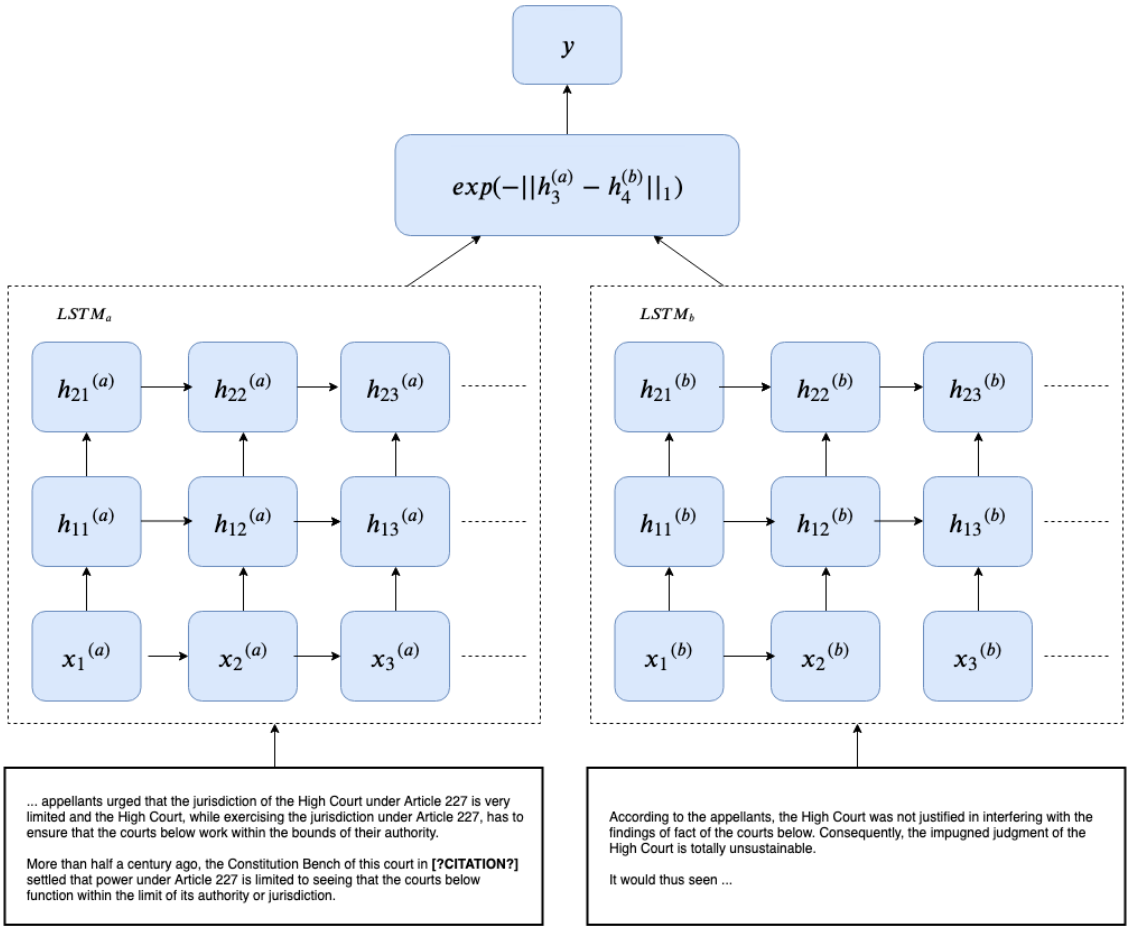

Figure 3. Architecture of Siamese networks using LSTM with Manhattan Distance

We had implemented our MaLSTM using Keras. The current case and precedent case candidate documents were preprocessed before passing them on to the LSTM layer. Tokenization, removal of stop words and lemmatization using the Wordnet Lemmatizer were performed on each of the documents before converting them to a format that the LSTM accepts. From the given dataset, we had a list of 200 current cases and a pool of 2000 precedent cases. Each of the 200 current cases had been labelled with the list of precedents. In order to prepare the input vectors, we started with an empty map. Then for each pair of the current case and its precedent, an entry was added into the map such that the size of the resulting map was 200 × sum of the precedents for each of the current cases. This formed our positive datasets. Thereafter, for each of the 200 current cases, we selected up to 4 prior cases which were not the precedent of that case. These formed our negative data points. Each of the current and prior case pairs were converted into their corresponding word embeddings.

The inputs to the LSTM network were fixed-length zero-padded sequences of word indices, with each non-zero index uniquely identifying each word. This vector is then fed into the embedding

layer which fetches the embedding corresponding to each word and encapsulates all of them into a matrix. We have used Word2Vec [14] to retrieve the embeddings for each word.

The two embedded matrices, one representing the precedent case document and the other representing the precedent candidate case document are then fed into the LSTM networks as shown in Figure 3. The LSTM outputs a fixed dimensional hidden state vector which represents the semantic meaning of the input text. The similarity between the two hidden state vectors that hold the semantic meaning of the current case and the precedent case candidate is then computed using the Manhattan similarity function :

$$\exp(-\| h^{(left)} - h^{(right)} \|_1)$$

wherein $h^{(left)}$ and $h^{(right)}$ refer to the hidden state vectors output by the two LSTMs.

Since the exponent of a negative is being calculated, the output is always a floating-point integer between 0 and 1, representing the similarity score between the two documents. Additionally, we have used the Adam [28] optimizer to boost accuracy by finding better local minima.

5.2.1. Parameters and Hyperparameters

For our LSTM model, we trained the model with different parameters. We experimented with output dimensionality between a range of 50 – 100. Optimization of the parameters is done using Adam algorithm which is a stochastic gradient descent method that is based on adaptive estimation of first-order and second-order moments. We used this algorithm since the method has a little memory footprint and is computationally efficient [28]. We did not use any specific scheme in order to initialize our weights but used random weight initialization for our model.

The three important hyperparameters used in our model are – embedding dimensions for word embeddings, maximum length for the sequences with padding, and dimensionality of the output space. The results with different values of these hyperparameters are documented in section 7 of our paper.

6. EVALUATION

We have evaluated all our models using the popular Information Retrieval metrics that have been used by most other works [3] as well. These metrics are:

1. Precision@10 [29]
    a. Precision@10 is the proportion of recommended items in the top 10 that are relevant

       Precision@10 = (no. of recommended items in top 10 predictions that are relevant) / 10

2. Recall@100 [29]
    a. Recall@100 is the proportion of relevant items present in the top 100 recommended items

       Recall@100 = (no. of recommended items in top 100 predictions that are relevant) / (total no. of relevant items)

3. Mean Average Precision [30]

a. Average precision is a measure that amalgamates precision and recall for ranked retrieval systems. For each retrieved list, average precision is the mean of the precisions computed after each relevant item that has been retrieved [29]. When such ranked lists are to be generated for multiple entities, the mean of their average precisions forms the Mean average precision score, which is a single score representing the retrieval efficiency across all the entities.

$$Average\ Precision = \sum_{i=1}^{n} (precision@i \times change\ in\ recall\ at\ i)$$

4. Mean Reciprocal Rank [31]
   a. The Reciprocal Rank measure computes the reciprocal of the rank at which the first relevant document was retrieved [30]. When such ranked lists are to be generated for multiple entities, the mean of their reciprocal ranks forms the Mean reciprocal rank score, which is a single score representing the retrieval efficiency across all the entities.

$$MRR = \frac{1}{n} \sum_{i=1}^{n} \frac{1}{rank\ of\ i}$$

7. RESULTS AND DISCUSSION

Table 1. Effectiveness of Wordnet based precedent case retrieval system

	MAP	MRR	Precision@10	Recall@100
Wordnet with Leacock Chodorow similarity measure	0.419	0.836	0.233	0.793
Wordnet with Jiang-Conrath similarity measure	0.477	0.801	0.26	0.789
flt_ielab_idf [12]	0.390	0.719	0.236	0.781

Our Wordnet based precedent retrieval system leveraging the Jiang-Conrath similarity measure to evaluate the similarity between two synsets outperformed that using the Leacock Chodorow similarity measure in terms of precision i.e. for both the mean average precision and the precision@10 metrics as depicted in Table1. On the other hand, the variant using the Leacock Chodorow similarity measure performed better than the variant using the Jiang-Conrath similarity measure in terms of the mean reciprocal rank and recall i.e. the recall@100 metric. We state this explicitly as the overall recall is also an important metric as a lawyer might prefer to go through a large number of precedent case candidates than just of few of them due to the fear of missing out on some important potential citations [3].

Both the variants performed significantly better after lemmatizing the queries and the precedent case candidate documents before comparing them. Additionally, the concept of using the paragraph containing the citation markers as the queries boosted the efficiency of our system as against comparing the current and precedent case documents in their entirety.

Our variant using the Jiang-Conrath similarity measure outperformed the flt_ielab_idf baseline we had considered [12] in terms of all metrics whereas the variant using Leacock Chodorow similarity measure outperformed the flt_ielab_idf baseline [12] in all metrics except the precision@10.

In our second approach, where we have used the Siamese Manhattan LSTM, we have experimented with different variations by varying the values associated with the different

hyperparameters while training our LSTM model. However, the conceivable difference was minute. The different hyperparameters are:

- *Dim:* embedding dimension for Word2Vec
- *Maxlen:* max length for all the sequences when padding the training set
- *units :* dimensionality of the output space

Table 2. Effectiveness of MaLSTM based precedent case retrieval system

	MAP	MRR	Precision@10	Recall@100
Variation 1	0.111	0.445	0.108	0.322
Variation 2	0.082	0.046	0.333	0.267
Variation 3	0.121	0.567	0.112	0.206
Variation 4	0.135	0.617	0.126	0.320
Variation 5	0.103	0.442	0.1	0.28

Variation 1 : *dim = 300, maxlen = 60, units = 50*
Variation 2 : *dim = 300, maxlen = 60, units = 100*
Variation 3 : *dim = 2000, maxlen = 60, units = 100*
Variation 4 : *dim = 1000, maxlen = 100, units = 100*
Variation 5 : *dim = 300, maxlen = None, units = 500*

The best result we could achieve with LSTM was a mean-average-precision of approximately 0.135. The poor performance of the models can be attributed to the unavailability of a large enough dataset to effectively train our model. The Siamese Manhattan LSTM had performed very well on sentence similarity applications where the model had been trained on several thousands of sentences [20]. Given that our dataset consisted of only 200 current case documents, we trained our model on 80% of the cases, i.e. on 160 current cases and used the remaining 40 cases as part of our test set. We believe that it has performed reasonably well with such a small training dataset.

8. FUTURE ENHANCEMENTS

The Siamese Manhattan LSTM based precedent case retrieval system relies heavily on the size of the training dataset. Though the Siamese Manhattan LSTM based precedent case retrieval system has not performed as well for some of the other approaches discussed earlier, we do find the results encouraging enough to revisit this approach and re-evaluate it in the future once we have access to a much larger dataset. We have used Word2Vec [14] and Doc2Vec [15] for converting the documents to their respective word embeddings. We would also experiment with the fastText [32], which is a library for the learning of word embeddings created by Facebook's AI research lab. For the purpose of creating the word embeddings, we used our dataset as the corpora for training Word2Vec. We would like to experiment with embedding models trained specifically on huge legal data sets such as Law2Vec [33] and the Word2Vec and Doc2Vec legal models provided by LexNLP[34].

9. CONCLUSION

In this paper, we have proposed two new approaches to solve the information retrieval problem of precedent case retrieval. Our approach leveraging Wordnet as a knowledge-based measure to quantify the semantic relatedness between words using a semantic network achieved a good mean average precision and a high recall rate. This approach has outperformed all the existing precedent case retrieval methods evaluated on the same dataset as that as we have used. Our second approach involving a Siamese Manhattan LSTM performed poorly as it was trained on a minuscule set of data points. However, the results were promising enough to see the potential for it to perform efficiently when trained on a large enough dataset.

REFERENCES

[1] Chemin, M., 2010. Does court speed shape economic activity? Evidence from a court reform in India. The Journal of Law, Economics, & Organization, 28(3), pp.460-485.

[2] 3.3 crore cases pending in Indian courts, pendency figure at its highest: CJI Dipak Misra

[3] Mandal, A., Ghosh, K., Bhattacharya, A., Pal, A. and Ghosh, S., 2017. Overview of the FIRE 2017 IRLeD Track: Information Retrieval from Legal Documents. In FIRE (Working Notes) (pp. 63-68).

[4] https://en.wikipedia.org/wiki/Common_law/

[5] Mandal, A., Chaki, R., Saha, S., Ghosh, K., Pal, A. and Ghosh, S., 2017, November. Measuring similarity among legal court case documents. In Proceedings of the 10th Annual ACM India Compute Conference (pp. 1-9). ACM.

[6] Kumar, S., Reddy, P.K., Reddy, V.B. and Singh, A., 2011, March. Similarity analysis of legal judgments. In Proceedings of the Fourth Annual ACM Bangalore Conference (p. 17). ACM.

[7] Working Notes : On the importance of legal catchphrases in precedence retrieval.

[8] Raghav, K., Reddy, P.K. and Reddy, V.B., 2016. Analyzing the extraction of relevant legal judgments using paragraph-level and citation information. AI4JCArtificial Intelligence for Justice, p.30.

[9] Thenmozhi, D., Kannan, K. and Aravindan, C., 2017. A Text Similarity Approach for Precedence Retrieval from Legal Documents. In FIRE (Working Notes) (pp. 90-91).

[10] Sandeep, G.V. and Bharadwaj, S., 2017. An Extraction based approach to Keyword Generation and Precedence Retrieval: BITS Pilani-Hyderabad. In FIRE (Working Notes) (pp. 74-77).

[11] Kulkarni, Y.H., Patil, R. and Shridharan, S., 2017. Detection of Catchphrases and Precedence in Legal Documents. In FIRE (Working Notes) (pp. 86-89).

[12] Locke, D. and Zuccon, G., 2017. Automatic cited decision retrieval: Working notes of Ielab for FIRE Legal Track Precedence Retrieval Task.

[13] G. Salton and C. Buckley. Term weighting approaches in automatic text retrieval. Technical report, 1987.

[14] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. CoRR abs/1301.3781 (2013).

[15] Quoc Le and Tomas Mikolov. 2014. Distributed Representations of Sentences and Documents. In Proc. International Conference on Machine Learning (ICML), Tony Jebara and Eric P. Xing (Eds.). JMLR Workshop and Conference Proceedings, 1188–1196.

[16] Reshma, U., Kumar, M.A. and Soman, K.P., 2017. Distributed Representation in Information Retrieval-AMRITA_CEN_NLP@ IRLeD 2017. In FIRE (Working Notes) (pp. 69-71).

[17] Miller, G.A., 1995. WordNet: a lexical database for English. Communications of the ACM, 38(11), pp.39-41.

[18] Hochreiter, S. and Schmidhuber, J., 1997. Long short-term memory. Neural computation, 9(8), pp.1735-1780.

[19] Chung, J., Gulcehre, C., Cho, K. and Bengio, Y., 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint arXiv:1412.3555.

[20] Mueller, J. and Thyagarajan, A., 2016, March. Siamese recurrent architectures for learning sentence similarity. In Thirtieth AAAI Conference on Artificial Intelligence.

[21] Porter, M.F., "An algorithm for suffix stripping," Program, vol.14, pp. 130-137. 1980.

[22] Hooper, R. and Paice, C., 2005. The Lancaster stemming algorithm. University of Lancester.

[23] Leacock, C. and Chodorow, M., 1998. Combining local context and WordNet similarity for word sense identification. WordNet: An electronic lexical database, 49(2), pp.265-283.

[24] Lee, S.hyun. & Kim Mi Na, (2008) “This is my paper”, ABC *Transactions on ECE*, Vol. 10, No. 5, pp120-122.

[25] Wu, Z. and Palmer, M., 1994, June. Verbs semantics and lexical selection. In Proceedings of the 32nd annual meeting on Association for Computational Linguistics (pp. 133-138). Association for Computational Linguistics.

[26] Resnik, P., 1995. Using information content to evaluate semantic similarity in a taxonomy. arXiv preprint cmp-lg/9511007.

[27] Jiang, J.J. and Conrath, D.W., 1997. Semantic similarity based on corpus statistics and lexical taxonomy. arXiv preprint cmp-lg/9709008.

[28] Kingma, D.P. and Ba, J., 2014. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

[29] Zeiler, M.D., 2012. ADADELTA: an adaptive learning rate method. arXiv preprint arXiv:1212.5701.

[30] Herlocker, J.L., Konstan, J.A., Terveen, L.G. and Riedl, J.T., 2004. Evaluating collaborative filtering recommender systems. ACM Transactions on Information Systems (TOIS), 22(1), pp.5-53.

[31] Craswell, N., 2009. Mean reciprocal rank. Encyclopedia of Database Systems, pp.1703-1703.

[32] Joulin, A., Grave, E., Bojanowski, P. and Mikolov, T., 2016. Bag of tricks for efficient text classification. arXiv preprint arXiv:1607.01759.

[33] Ilias Chalkidis. Law2Vec: Legal Word Embeddings.

[34] Bommarito, I.I., Michael, J., Katz, D.M. and Detterman, E.M., 2018. LexNLP: Natural language processing and information extraction for legal and regulatory texts. arXiv preprint arXiv:1806.03688.

AUTHORS

Sai Vishwas Padigi received his Bachelor of Engineering degree in Computer Science and Education from PES Institute of Technology, Bangalore, India. His areas of research interest include data mining, NLP, artificial intelligence, machine learning , sustainable computing and cloud computing.

Mohit Mayank received his Bachelor of Engineering degree in Computer Science and Education from PES Institute of Technology, Bangalore, India. His areas of research interest include information retrieval, deep learning and cloud native technologies.

S Natarajan is a Ph.D. from JNTU, Hyderabad in the domain of Remote Sensing. He has a vast experience of 33 years in R&D (at Defence Research and Development Laboratory (DRDL) and National Remote Sensing Agency (NRSA)) and more than 15 years in teaching. He is currently working as Professor in Computer Science and Engineering department at P E S University, Bangalore, Karnataka, INDIA. He has served as conference chair, reviewer and session chair for various international conferences and journals. At ISRO he had handled project management for Satellite Acquisition and Processing System for Defense Electronics Applications Laboratory. He has delivered tutorials on Spatial Data Structures, Geographic Database generation, and Geographic Information Systems during International Conference on Remote Sensing and GIS. He has received ACM best paper and ACM best poster awards. He has more than 100 publications to his credit. He is a member of CSI, ISRS, INCA, and ISTE.