Artificial Intelligence and Law (2021) 29:417–451
https://doi.org/10.1007/s10506-020-09280-2

ORIGINAL RESEARCH

Unsupervised approaches for measuring textual similarity between legal court case reports

Arpan Mandal[1] · Kripabandhu Ghosh[2] · Saptarshi Ghosh[3] · Sekhar Mandal[1]

Accepted: 14 December 2020 / Published online: 4 January 2021

Abstract
In the domain of legal information retrieval, an important challenge is to compute similarity between two legal documents. Precedents (statements from prior cases) play an important role in *The Common Law system*, where lawyers need to frequently refer to relevant prior cases. Measuring document similarity is one of the most crucial aspects of any document retrieval system which decides the speed, scalability and accuracy of the system. Text-based and network-based methods for computing similarity among case reports have already been proposed in prior works but not without a few pitfalls. Since legal citation networks are generally highly disconnected, network based metrics are not suited for them. Till date, only a few text-based and predominant embedding based methods have been employed, for instance, TF-IDF based approaches, Word2Vec (Mikolov et al. 2013) and Doc2Vec (Le and Mikolov 2014) based approaches. We investigate the performance of 56 different methodologies for computing textual similarity across court case statements when applied on a dataset of Indian Supreme Court Cases. Among the 56 different methods, thirty are adaptations of existing methods and twenty-six are our proposed methods. The methods studied include models such as BERT (Devlin et al. 2018) and Law2Vec (Ilias 2019). It is observed that the more traditional methods (such as the TF-IDF and LDA) that rely on a bag-of-words representation performs better than the more advanced context-aware methods (like BERT and Law2Vec) for computing document-level similarity. Finally we nominate, via empirical validation, five of our best performing methods as appropriate for measuring similarity between case reports. Among these five, two are adaptations of existing methods and the other three are our proposed methods.

Keywords Legal information retrieval · Court case reports · Court case similarity · Topic modeling · Word2vec · Doc2vec · BERT · Law2vec

This manuscript is an extended version of our prior work Mandal et al. (2017) "*Measuring Similarity among Legal Court Case Documents*", ACM COMPUTE conference 2017.

✉ Arpan Mandal
amarnamarpan@gmail.com

Extended author information available on the last page of the article

1 Introduction

One of the most followed legal systems in the world is the Common Law System. It is followed by one-third of the world's population in several countries including India, Australia, USA, and the United Kingdom[1]. An essential component of the Common Law System is that immense significance is imparted to *prior cases* and are considered to be as significant as some other written law in the statutes. A principle (called *Stare decisis*) in Common Law System decrees that comparable cases or related issues should earn comparable outcomes.[2] It suggests that if any prior case is similar to the situation being handled, the judicial bench is obliged to examine and adopt the interpretation of the relevant issue(s) from the previous case. A precedent is such a case which has been cited in the court of law. As precedents are considered to be equally influential to any other law in the statutes, lawyers often are required to identify these similar previously judged cases which are admissible in the court of law. This task of finding similar precedents are majorly done manually by the legal experts themselves. This manual labour is costly in the course of a litigation process which requires expertise as well as time.

The recent advent in Information Technology has led to an accelerated growth in the collection of digitally accessible legal documents. With a widened number of the legal documents, it is becoming more challenging for legal professionals to manually find similar precedents that would support a current case in progress. It is hence beneficial for legal professionals to have strategies that automatically retrieves similar precedents. Moreover, with an assistance of a precedent retrieval system, any lawyer can achieve an edge over the case.

To this end, finding similarity among the ongoing case and a previously decided case is a critical challenge of *computing the similarity between two court case documents* that needs to be addressed. Considering the fact that court case reports are generally lengthy and obscure in structure (Brüninghaus and Ashley 2001), this task is especially challenging. Additionally, analysis of several different legal issues might be included in a single legal case document. These complexities imply that expensive domain expert knowledge is needed to manually understand similarity between two documents. Hence, it is of special importance to automate the system of similarity measurement between two court case reports. The above mentioned constraints also pose the challenging nature of the work.

Among the earlier works, a few procedures for estimating similarity over two legal court cases have been proposed. The two significant kinds of these earlier works include: network-based techniques and text-based strategies. Network based methods rely upon the citation network between the court case documents, whereas Text-based methods rely upon the frequency based metrics of the given textual data. The earlier works also include a form of hybridization of network and text-based methods (see Sect. 2 for a thorough examination of these techniques). However, these prior methods have certain curcumstantial limitations. To check, how

[1] https://en.wikipedia.org/wiki/List_of_national_legal_systems/.

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

the network based methods perform we built our own citation network, which later turned out to be impractical to be used due to its sparsity, and the network-based measures can hence be not meaningfully computed for most of the case documents. This sparseness in legal citation network is also confirmed by (Kumar et al. 2011). Hence these methodologies are not always applicable to court case documents. Then again, as far as anyone is concerned, the content based techniques that have been applied on legal documents are for the most part crude, such as TF-IDF based approaches (Kumar et al. 2013).

Recent advancements in information technology has provided us with supervised neural models for measuring similarity between documents such as one work by (Liu et al. 2019) measures similarity between news articles. However, these supervised neural models require large quantities of training data (pairs of similar and dissimilar documents typically in the range of hundreds of thousands). Getting such large amounts of training data is prohibitively expensive in the legal domain, since judging similarity of legal documents requires involvement of law experts, which is both expensive and time-consuming. Hence, we focus on unsupervised approaches which do not need to be trained on similar and non-similar document pairs. These pitfalls of the earlier techniques prompted us to study more advanced methodologies for measuring the similarity between court case reports which rely just on the unsupervised methods of measuring document similarity.

In this work, we examine 56 different techniques (including 26 novel techniques) to compute similarity among a pair of court case documents. This is done by first recognizing different diverse forms of text representations of a case document, and then computing similarities among these text representations by several recent methodologies. Considering all the various available text representations of a case document, we examine (i) the whole document, (ii) the set of sentences of the document, (iii) the set of paragraphs in a document, (iv) summaries of the document, (v) Catchphrases extracted from the document and, (vi) the set of RFCs (the text surrounding a citation) from the document. The representations attempt to identify the significant/important parts of the case reports. Among these methods of representing the text, three methods have not been used as a text representation in any other prior work – *(a) Catchphrases using CRF(Conditional Random Fields), (b) Summarization using MyScore and (c) Summarization using PScore*.

The subsequent step is to compute the similarity between the report texts (or the textual representations). A conventional approach for this step is to design a vector representation of the document and hence calculate the cosine distance between the vectors. As mentioned before, earlier works (Kumar et al. 2013) utilized TF-IDF vectors for this reason. Vector representations are also applied in this work, for example, topic modeling (Latent Dirichlet Allocation (Blei et al. 2003) and neural network based embeddings (e.g., word embeddings by (Mikolov et al. 2013), document embeddings by (Le and Mikolov 2014) and BERT by (Devlin et al. 2018)). In addition, we have also used a pre-trained model of word embeddings called *Law2Vec* (available online at https://archive.org/details/Law2Vec) specifically trained on a large set of legal documents. The advanced strategies (Word2Vec, Law2Vec, Doc2Vec, BERT) capture the *semantics* of the case texts. Therefore these can identify similar case reports even if they use dissimilar terms in the text. We also adapt a pre-existing word scoring method (called *PScore*) as a method of measuring similarity across texts (called *PScoreVect*).

In this paper, we perform extensive experiments over a set of 56 different (26 novel) methods of measuring similarity among court case documents. For the sake of comparison, the baseline is considered to be the prior work by (Kumar et al. 2013), that first applies their technique upon 50 pair of documents and, hence correlates the similarity scores given by their technique with that given by the legal experts. We compare our methods with the same set of expert-given scores. While comparing among the different methodologies of measuring similarity, it is found that the embeddings learnt by the neural network techniques (Word2vec, Doc2vec, Law2Vec) perform comparably with the other techniques for the stated task. In fact, we see that BERT (Devlin et al. 2018) yields substandard results for the similarity estimation task, when trained on the dataset of 33, 500 Indian Supreme Court case documents. We experimented by training a BERT both from scratch as well as by fine-tuning it over the existing model. In both cases, the observed performance of BERT is inferior to that of the more traditional techniques (which can be due to the training set not being large enough). Overall, we observe Doc2vec embeddings over the entire document to be especially effective in estimating similarity among the court case documents.

The remainder of the paper is structured as follows. In Sect. 2, we quickly review the existing methods of computing similarities between court case reports. Section 3 describes the dataset that we use. Section 4 examines in detail the techniques that we have tried to measure similarity with. The performance of the various procedures is assessed in Sect. 5, which contrasts the proposed strategies and the baseline standard. Section 6 concludes our present work and mentions a few potential future works.

2 Related work

Prior works have attempted to measure similarity between dynamic documents in a highly scalable way by (Santos et al. 2011). These methods are useful and specifically built for systems where documents are prone to periodically getting modified. Hence, this method is not suited for our work. Attempts for measuring similarity between words has also been made using multiple ontological structures by (Batet et al. 2013). Ontologies are logical structure between the words of a single document or corpus. But in our work, instead of words we are trying to find similarity between two documents, specifically legal documents.

The techniques for estimating similarity between two court case reports can be largely categorized into *supervised* techniques, *text-based unsupervised* techniques, and *network-based unsupervised* techniques. Likewise, some combined approaches have utilized a mix of text-based and network-based techniques. In this section, we shortly talk about existing methodologies of estimating similarity between legal reports. We also show why the recent AI-based strategies are not appropriate for estimating similarity between legal documents.

2.1 Supervised similarity measuring methods for legal and other documents

Recent advancements in the field of Machine Learning has availed many supervised methods for measuring document similarity. Two of the most recent ones include works by (Ahmad et al. 2018), and (Chen et al. 2018). The first uses transfer learning to learn sentence embeddings using annotations from different sources to create an amalgamated sentence embedding using BLSTMs (bidirectional Long short term memory units – a form of artificial neural network architecture). The second uses a custom built CNN (Convolutional Neural Network) model on top of pre-existing embedding models to correctly predict if or not two text snippets are similar.

Although there has been many works on measuring similarity between generic text documents, not many works have been done which are specifically applicable on the legal documents. To the best of our knowledge the only existing work on legal documents is by (Sugathadasa et al. 2018). They form an ensemble of two pre-existing variants of Doc2Vec (Node2Vec-based and Sentence-Similarity-based) to propose a newer improved Doc2Vec model. This work uses supervision to fine-tune an unsupervised document embedding technique. The supervised part of their method requires example pairs of similar and dissimilar legal documents. For this, they have used a dataset of 2,500 documents that had been crawled from the web and is publicly unavailable.

Despite the progress, there still exists a fundamental bottleneck in applying supervised methods of similarity measurement over *legal documents*. Building enough training data for a supervised method of similarity measurement is a costly process, as it requires legal experts to manually analyze the document text and assign a degree of similarity for a given pair of documents. For this practical reason, this work exclusively explores the unsupervised ways of measuring document similarity among legal documents.

2.2 Unsupervised similarity measuring methods for legal and other documents

These methods are essentially language modeling tools that can convert a given text into numerical vectors so that when given two documents, they can just be converted into two numerical vectors and measure similarity among the two vectors using some form of distance between them (cosine/euclidean). The most notable among these is the *Word2Vec* model (Mikolov et al. 2013) that can effectively convert a given word into a numeric vector while preserving its semantic properties. We have adapted Word2Vec as a method for similarity measurement as described in Sect. 4.2.5. A pre-trained legal version of the Word2Vec model known as Law2Vec is available online (Ilias 2019). This model is pre-trained upon 123,066 documents aggregated from 8 different sources. We have also experimented using the pre-trained Law2Vec embeddings.

The topic modeling technique *LDA* or *Latent Dirichlet Allocation* (Blei et al. 2003) is another notable method where we can assign a fixed number of topics and when trained over a corpus of text documents, it automatically assigns a belongingness value over each abstract topic. We have used this technique in our experiments and the method is described later in the paper.

Apart from these pre-dominant methods we have other state-of-the-art methods which require very large corpus of texts to generalize the model. Examples include, BERT (Devlin et al. 2018), and ELMo (Peters et al. 2018). Among these we have tried BERT as a mechanism to measure similarity among legal documents. Understanding the inner workings of BERT requires knowledge of Deep Learning and Artificial Neural Networks, and is hence best explained by their original paper. The way we adapted BERT for our problem is explained briefly later in the paper.

2.3 Network-based similarity measures for legal documents

The principal element in a network-based similarity measuring system is a citation network of legal case reports. A citation network of legal case reports is a directed graph where, each of the nodes is a case report, and each of the directed edges $A \rightarrow B$ between two records connotes that the record A refers to the report B. For example, B may be an earlier case that is referred to as a citation by a newer case A.

Different similarity methods have been introduced built upon the case reports citation network. For example, two network based measures were proposed by (Kumar et al. 2011) - *bibliographic coupling* and *co-citation*. Bibliographic coupling among two reports is the quantity of basic 'out-references' of these two archives. While, 'co-citation' is the quantity of common 'in-references' between the two archives. It was demonstrated that the two network-based techniques are suitable for identifying similar decisions.

One work by (Minocha et al. 2015) have proposed two other network-based similarity measures - *dispersion* and *embeddedness*. *Dispersion* is a well-established measure that has been utilized on Facebook networks (Backstrom and Kleinberg 2014) to anticipate the connection of romantic relations inside a miniature social network. When two nodes are provided, the Dispersion metric tries to compute the degree are the intersection of neighbours between the nodes (*excluding themselves*) all around are associated. The other measure, Embeddedness, in straightforward terms implies how probable the regular neighbours of a node are to shape a clique.

In spite of the fact that these techniques perform well in identifying similar case documents, clearly the applicability of these strategies relies upon how well-coupled the fundamental citation graph is. Court case reports, for the most part, give rise to an exceptionally sparse citation graph as was shown by (Kumar et al. 2011). For example, in the dataset used for this paper, the citation graph is so sparse that 80% of the nodes were found to be isolated nodes (i.e., neither did they cite any paper, nor were they cited by any other). The sparseness of legal citation networks is essentially due to the fact that legal counsellors/judges usually do not note down all conceivable cases pertinent to a given case; rather, just a couple of the most significant earlier cases are cited in the case report text. In this scenario, the cases that are already cited by a more recent case is likely to be cited repeatedly to handle similar situations. In such conditions, text-based relevance metrics (or hybrid ones) are more helpful in searching for unexplored (but relevant cases) than simply suggesting a well-known (but not so relevant) case via network-based metrics.

2.4 Text-based and hybrid similarity measures for legal documents

Apart form the network-based measures reviewed earlier, (Kumar et al. 2011) proposed two text-based measures – all-term cosine similarity, and legal-term cosine similarity.

The two methods depend on the TF-IDF scores of words present in the report text. The reports are expressed as vectors, of the size equivalent to the total count of distinct words in the dictionary of the whole text corpus. Along these lines, every element/dimension of the vectors would relate to a distinct word in the dictionary and carries the TF-IDF score of the word. When the vectors for both the reports are composed, cosine distance is measured between the two vectors to get the similarity between the text reports.

In addition to this, a hybrid method was further introduced by the same authors (Kumar et al. 2013) that took into account both the textual content as well as the citation network information. In this method, they presented a measurement called *paragraph-links* or PLs. To compute PLs, the text of each report is divided into a collection of constituent paragraphs. Hereafter, each paragraph is considered as a separate text segment. Hence a similarity score over every set of paragraphs is estimated. Then, in the event that the similarity between any two paragraphs is higher than a threshold, at that point a *paragraph link* is included between the two reports that contains the pair of similar paragraphs. Strictly speaking, let $[A_1, A_2, A_3, ...A_i]$ be the arrangement of paragraphs in report A and likewise let $[B_1, B_2, B_3, ...B_j]$ be the arrangement of paragraphs in report B. In this way, by measuring similarity among the constituent paragraphs of A and B, we have $i \times j$ similarity scores. Finally, a *paragraph-link* is embedded among A and B if any of these $i \times j$ similarity scores is higher than a threshold.

For computing likenesses between two paragraphs, they originally calculated TF-IDF vectors for the paragraphs (considering the constituent words of the two paragraphs), and afterwards calculated the cosine similarity between these two vectors. They demonstrated that the performance of PL strategy exceeded over that of the bibliographic coupling measure (Kumar et al. 2011), when evaluated upon a limited set of 3,866 Court Case Judgments from the Indian Supreme Court. This study (Kumar et al. 2013) was considered as a baseline in this work.

2.5 Present work as an extension of our prior work

In our previous work (Mandal et al. 2017a), we had experimented with several methodologies for similarity measurement across legal case reports, including neural network models such as word-level (Word2Vec) and document-level embeddings (Doc2Vec) as well as text-based models such as topic models (LDA), and TFIDF. In addition to these, in this work we have experimented with three more embedding techniques:

1. *PScoreVect*: *PScore* is a word scoring function as described by (Mandal et al. 2017c). This scoring function is adapted as a technique for finding document embeddings The method is described later in Sect. 4.2.3
2. *Law2Vec* (Ilias 2019): A Word2Vec model pre-trained on several legal documents from various sources.
3. *BERT* (Devlin et al. 2018): This is the state-of-the-art in Language modeling and can also generate embeddings for a given text.

In the previous work, we had experimented using just four representations of document – Whole Documents, MyScore Summaries, RFCs, and Paragraphs. In the present work, along with these four we have experimented with four more document representations:

1. *PScore Summary*: This is the top ten percent of the highly scored sentences from a document when scored using the *PScore* scoring function as described later in Sect. 4.1.2. *PScore* is a scoring function taken from a work by (Mandal et al. 2017c).
2. CRF catches: A set of catchphrases from a given document using *CRF* or *Conditional Random Fields*. *CRF* is a supervised model for the task of Sequence tagging. This is used to extract important phrases from the document. The set of these important phrases can then be used as a representation of the document itself.
3. PS Catches: A set of catchphrases extracted for a given document using the *PScore* scoring function. This is the same scoring function that is used in case of *PScore Summary*.

Note that, the total number of methods for measuring document similarity that we experiment with is the number of document representations times the number of ways we convert the text representation into vectors. In our earlier work the total number of methods were sixteen ($4 \times 4 = 16$). In our present work, the total number of methods that we experiment with is fifty six ($8 \times 7 = 56$). Apart from an increased number of methods that we experiment with, we have considered a new method of evaluating our methods. In contrast to our prior work, where we considered just the *Pearson Correlation Coefficient* as the measure, we additionally consider *Accuracy* as another measure of evaluation in our present work. We perform experiments over the same set of 47 pairs as was done earlier in our previous work (Mandal et al. 2017a) and by our baseline paper (Kumar et al. 2013), and finally find five different methods tried in this work that substantially outperform the approaches in prior works.

3 Datasets

In this section we briefly describe the two non-intersecting sets of datasets used. One dataset was used to train the unsupervised models of similarity measurement the exhaustive list of which includes – Word2Vec, Doc2Vec, TFIDF, LDA, BERT

Table 1 This table gives an overview of the two datasets used in this work. The training dataset is used for training the self-supervised algorithms such as Word2vec, Doc2vec. The evaluation dataset is used for evaluating the performance of different methods

Training dataset	
Nos. of documents	33,545
Average document length (number of words)	3,829
Standard deviation of document lengths	5,485
Evaluation dataset	
Nos. of document-pairs	47
Range of expert scores	[1, 10]
Average of expert scores	4.26
Standard deviation of expert scores	3.61

and PScoreVect. Another dataset is a collection of case report pairs, for which the expert similarity scores are provided. They are briefly described below.

3.1 Dataset of legal documents for training models

In this work we use several techniques of converting texts into vectors (TF-IDF vectorizer, LDA, Word2Vec, Doc2Vec, and BERT) which needs to be generalized on a corpus of legal texts. To this end, a collection of 33,545 reports of court cases from the Supreme Court of India was selected. The collection comprises of all legal case judgements from the Supreme Court of India, spanning over a course of 67 years (since 1950 till 2016), obtained as simple text documents. The main properties of the dataset are given in Table 1. The case reports were collected from an online website called the *LIIofIndia*. (http://www.liiofindia.org/databases.shtml), which is a website facilitated by the Legal Information Institute of India, that transparently hosts an assortment of law-related databases.

3.2 Gold-standard similarity scores for case reports (for evaluation of methods)

To understand the efficacy of our strategies through comparison, we required a Gold-standard level of expert-given scores for two given case reports. Although hiring some legal expert to get us some scores on a collection of pairs of documents would have been ideal, this solution was a costly one. Instead we relied upon an already provided expert scores from one of the prior works in this field, as a more feasible and effective solution. The prior work by (Kumar et al. 2013) provides 50 pairs of case reports for which they also provide the legal expert scores. For this, we have utilized a collection of 50 sets of case reports from the above-mentioned earlier work (which acts as our baseline, discussed in Sect. 5.1). Among the 50 test pairs of cases, there were three such pairs for which in any event one of the reports was absent in our dataset as the corresponding case reports could not be crawled from the *LIIofIndia website*. Consequently, it was not feasible to estimate similarity for these three pairs of documents and therefore, we consider expert similarity scores for the leftover 47 pairs of cases as our gold-standard data. For every one of these pairs, the specialists were asked to minutely study the cases and accredit some relevance within the range of 0 to 10, where 0 shows the least relevance, while a score of 10 shows exceptionally similar reports. Table 6 states the report pairs (column heading - *Case Pairs*), alongside the expert scores (gold-standard) (column heading - *Expert Scores*) as detailed in (Kumar et al. 2013). The main properties of the dataset are given in Table 1. Note that this test set of 47 pairs of documents is fully disjoint from the larger dataset used for training the models.

4 Experiments on measuring document similarity

As mentioned before, in this report, we explicitly consider text-based procedures for calculating the similarity of two legal reports. All the experiments that are preformed in this work has two sub-steps for estimating relevance between the textual content of two legal case reports:

1. **Finding a suitable representation of the document text** - most ordinarily, the representation can be the entire text of the report. However, it is reasonable and probable that a legal case report might discuss many distinct legal issues in the same document. Consequently, a different methodology can be to choose suitable text segments of the report (with the end goal that every segment apprehends an individual legal issue), and afterwards to quantify the similarity between the segments of the two reports.
2. **Measuring similarity among different representations of a given pair of documents**—when we are provided with the chosen representations of the two reports, an appropriate technique should be utilized to calculate the similarity between the representations.

For the first step, there are seven ways of representing a text, and for the second step the similarity can be measured in any of the eight ways. So as portrayed in Fig. 1, we have performed 56 different experiments in this work.

Next, we explain in detail the procedures under the two sub-steps that we experimented with. Various approaches to choose the representation of a case report are explained in Sect. 4.1, while the next Sect. 4.2 talks about different approaches to quantify closeness between the representations. Among these, three of the ways to choose a representation of the document is our novel approach and one of the methods of measuring similarity among these representations is a novel technique. All others are an adaptation of some pre-existing techniques for the task of measuring similarity across court case documents.

4.1 Different representations of a legal case report

We employ basic approaches to either divide the entire text into fragments, or extract meaningful segments from it. In the following subsections, we provide brief outlines of the fragmentation/extraction strategies that we examined.

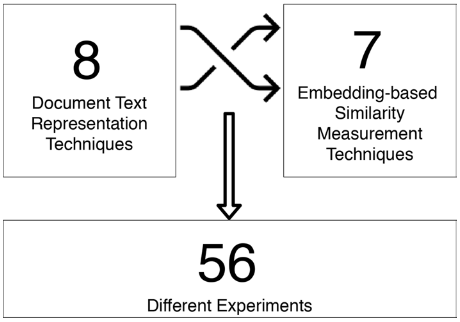
8
Document Text Representation Techniques
7
Embedding-based Similarity Measurement Techniques
56
Different Experiments

Fig. 1 Two steps in our experimental setup leads to $8 \times 7 = 56$ different experimentations in total

4.1.1 Whole document

The first and most straightforward system of all is perhaps to deal with the entire report text at once. Nonetheless, after a few discussions with the legal specialists, we inferred that an individual case report may argue and examine more than one legal issue. For example, a similar case report may talk about *legitimacy of an evidence*, *the seriousness of a wrongdoing committed* and *the apprehension of a statutory law*. At the point when a report d_2 is referred to as an earlier case from the case report d_1, it is conceivable that d_1 and d_2 share at least one legal issue for all intents and purpose (for which the reference was done), while the two reports might, by and large, not be fundamentally alike. To grasp such partial similarities between two case reports, we additionally examine the following different representations of a legal case report.

4.1.2 Summary of the document

The summary of a report gives us a short description of the main ideas talked about in the report while sifting through the excess, less significant parts. Consequently, utilizing summaries rather than the total content can naturally give us concise representations of the significant ideas that are talked about in a report.

To compose the summary of a case report, we devise our own summary extractors using two existing word-scoring functions in the following manner. A *significance score* is first computed for each sentence in the document. Hence, the sentences are arranged in descending order of their scores, and the highest 10% scored of all the sentences are chosen to compose the summary. In case of documents with less than 50 sentences, where 10% of the sentences were found to be under 5, at least five sentences were chosen to compose the summary. Choosing the size of the summary to be 10% of all sentences is purely a heuristic in our case. Choosing this value was inspired by another work of legal document summarization (Tran et al. 2020) where, it was shown by varying the percentage that 10% gave the best results in their work.

To determine the score for each sentence, we have used two different techniques:

1. a word scoring technique called *MyScore* was used as described by (Galgani et al. 2012).
2. another method of scoring words called *PS* as is described in (Mandal et al. 2017c).

To score a sentence by any of the two techniques, we initially compute a grade value for each of the *words* in the sentence and afterward, take a mean of these word-scores. The scoring of terms can be done using the two distinctive strategies as described below.

(1) MyScore While scoring with MyScore, only those words are considered to be terms, which occurs at least twice within the document (after stopword filtering, and stemming operations are performed over the text). For each term in the sentence, three unique scores are determined. The words are ranked by independently assigning them an aggregated score that is an amalgamation of three different scores. The words are separately ranked using each of the three scoring functions, and their resultant score is considered to be the mean of the position of the given word in the three different ranked lists. The three scoring functions utilized to sort the words are - (i) TF-IDF score of the word, (ii) frequency of the word in the document, (iii) the third score is calculated as follows. First, we structure an assortment of sentences that contain at any rate one of the ten most recurring words in the record. Let the set of such sentences be S. Next, we consider all terms t in the sentences in S that has a minimum of 2 occurrences in the document and call this set C. For every one of the candidate terms $t \in C$, we compute a proportion $TF(t, S)/TF(t, Doc)$, where, $TF(t, S)$ is the number of occasions t has appeared in S, also, $TF(t, Doc)$ is the number of occasions t has appeared in the Document.

Subsequently, a term-score is available for each term (that occurs more than twice in the document text), and derived from these term-scores, each sentence is attributed a score by finding the mean of scores of all terms present in the sentence. The summary is finally composed of 10% of the top-scored sentences

(2) PScore This is a frequency-based method for scoring words. This method gives importance to domain-specific terms by discriminating and differentiating between the probability distributions of a generic corpus of textual data and, a domain-specific corpus of legal documents. Although the method can be better understood in more detail in its original paper (Mandal et al. 2017c), we elucidate it briefly here. The scoring function can be mathematically represented using two equations:

$$Importance(t, \mathbb{C}) = \frac{CF(t, \mathbb{C})}{DF(t, \mathbb{C}) + 1} \tag{1}$$

where, $CF(t, \mathbb{C})$ is the collection frequency of the term t in corpus $\mathbb{C}$, and $DF(t, \mathbb{C})$ is the number of documents in $\mathbb{C}$ that the term t occurs in. and,

$$Score(t, \mathbb{C}_l, \mathbb{C}_{nl}) = \frac{Importance(t, \mathbb{C}_l)}{Importance(t, \mathbb{C}_{nl}) + 1} \tag{2}$$

where $\mathbb{C}_l$ and $\mathbb{C}_{nl}$ are the legal and non-legal corpus respectively. Regarding equation 2, it is notable that it is a fraction made up of two *Importance* values. The denominator is the importance of the term t in non-legal corpus and the numerator is the importance of the term in legal corpus. So, *Score* is the measure of how important a term is legally, and how unimportant it is in a non-legal corpus. In this work, we have used the famous *20 newsgroup dataset*[3] as the non-legal corpus.

4.1.3 Catchphrases extracted from a document

Catchphrases are single/multi-word phrases, a collection of which is specific to a legal document and, represents the gist of that document concisely. In this method, a document text is converted to its set of extracted legal catchphrases. To represent a document via a set of catchphrases, we use two forms of catchphrase extraction methods: (1) PS catches from a prior work by (Mandal et al. 2017c), and (2) CRF catches from another work by (Mandal et al. 2017b).

(1) PS catches PS catches is a way of extracting catchphrases from a given text document (Mandal et al. 2017c). It has two simple steps:

1. to extract noun phrases from a document
2. to score the noun phrases and choose top ten percent of them as catchphrases

To extract noun phrases we use a custom grammar and recursively extract noun phrases from each sentence as is detailed in our aforementioned paper. The grammar uses the Stanford POS tags (Toutanova et al. 2003) of the words to extract the noun phrases.

To score the noun phrases we use a method called *PS*. *PS* is a phrase scoring method which builds upon the word scoring function *Score* as mentioned in Sect. 4.1.2. To adapt this word scoring function as a phrase scoring function we consider frequencies and probabilities of phrases instead of words wherever required. *PS* can be mathematically represented as:

$$PS(c, \mathbb{C}_l, \mathbb{C}_{nl}) = \log \left[\sum_{i=1}^{|c|} Score(t_i, \mathbb{C}_l, \mathbb{C}_{nl}) \right] \cdot KLI(c, d) \tag{3}$$

where t_i is the i^{th} term of the noun phrase c and, $\mathbb{C}_{nl}$ and $\mathbb{C}_l$ are the non-legal and legal corpus respectively. So, *PS* is the product of two terms. The first is the *Score* as is described previously as a phrase scoring function and the other is the *KLI* or *KL divergence based informativeness* score. The *KLI* expression gives the explicitness of a term in a given case report, while the *Score* expression gives the particularity of the term in the legal domain. The KLI term can be calculated as follows:

[3] The 20 Newsgroups dataset can be obtained from https://archive.ics.uci.edu/ml/datasets/Twenty+Newsgroups.

$$KLI(c,d) = \frac{TF(c,d)}{|d|} \cdot \log \frac{\left(\frac{TF(c,d)}{|d|}\right)}{\left(\frac{CF(c)}{n}\right)} \tag{4}$$

where, $CF(c)$ is the number of times c has occurred in the whole corpus. The KLI score measures how well a noun phrase c represents a document d.

After the noun phrases are sorted we consider just top ten percent of the ranked noun phrases as the catchphrases of the document.

(2) CRF catches CRF or conditional random fields is a form of supervised sequence tagging model. When the model is fully trained using example training data, it can predict tags for the sequence elements. This has been used in many tasks, including named entity recognition and POS tag prediction (Pvs and Karthik 2006; Silfverberg et al. 2014; Luo et al. 2011; Ekbal et al. 2008), among others. CRF has been shown to perform well in case of legal catchphrase extraction, as was found by (Mandal et al. 2017b). For our purpose, we have used the CRF implementation as provided in *python-crfsuite*[4]. For training the CRF model we have used a separate set of documents for which the catchphrases are availed. To this end, we used a set of 400 documents along with their manually extracted legal catchphrases. The set of documents used for training the CRF model is hosted online at https://www.isical.ac.in/~fire/data/2017/IRLeD/FIRE2017-IRLeD-track-data.zip. The input format for the CRF model is a sequence of three entities – the tokenized word, the POS tag of the word, and its IOB label ('B' signifying beginning of a catchphrase, 'I' signifying intermediate or in-between a catchphrase, and 'O' signifying others), as shown in Table 2. This input is provided for training the CRF model. For prediction of tags, the tokenized form of participating documents (the two documents between which similarity is to be computed) are provided. Once the tags are predicted, a distinct set of corresponding phrases are extracted as the final set of catchphrases.

4.1.4 Paragraphs of the document

From our discussion with law practitioners, we understand that a particular legal case often examines several legal issues within the same document. But it is unclear as to how the legal issues can be separately identified.[5] Hence, we experimented with different levels of granularity which might possibly separate out areas of text that discusses a single legal issue. To this end, we tried breaking the text into either paragraphs or sentences.

We first discuss considering paragraphs of a document. Note that the strategy for recognizing a paragraph break in an archive is distinctly attributable to the corpus of texts. For our corpus, we broke the content into paragraphs in places where there

[4] **python-crfsuite** can be found online at https://python-crfsuite.readthedocs.io/en/latest/.

[5] Note that Indian Supreme Court case documents are, unfortunately, not divided into sections or subsections, which makes it even more difficult to identify the various legal issues or rhetorical sections in a document.

Table 2 Sample format of input data for conditional random fields (CRF) Above is a sample input format for the following snippet of text: "*The Vice-President shall not be a member of either House of Parliament or of a House of the Legislature of any State.*", when the following catchphrases are given: "*[vice-president, house of parliament, house of the legislature]*" For training the sequence taggers the text is first tokenized, the POS tags are predicted using stanford POS tagger and the labels are assigned. The labels are used to mark out the provided/predicted catchphrases. There are three labels: *B-tag, I-tag and O*. *B* stands for beginning, *I* stands for intermediate and *O* for others. During prediction, the same input format except the labels are presented to a trained model to get the labels

Word	POS	Label
the	DT	O
vice-president	JJ	B-tag
shall	MD	O
not	RB	O
be	VB	O
a	DT	O
member	NN	O
of	IN	O
either	DT	O
house	NN	B-tag
of	IN	I-tag
parliament	NN	I-tag
or	CC	O
of	IN	O
a	DT	O
house	NN	B-tag
of	IN	I-tag
the	DT	I-tag
legislature	NN	I-tag
of	IN	O
any	DT	O
state	NN	O

were two back to back newline characters. Additionally, we disposed of paragraphs that were shorter than 20 words long.[6]

It is apparent that estimating similarity via this strategy is not quite the same as the ones depicted previously. Here, a document d is fragmented into a number of paragraphs $\{p_1, p_2, ..., p_n\}$, where n is the total number of paragraphs in d. Given the two documents d_1 and d_2 between which the similarity is to be calculated, let n_1 and n_2 be the total number of paragraphs in d_1 and d_2 respectively. A total of $n_1 \times n_2$ similarity values can be determined among the paragraphs of these documents. To get the ultimate similarity score between the two case documents, we calculate the *mean* of all these similarity scores.

4.1.5 Sentences of the document

Sentences are a major building block of any textual document. In this representation, the document is broken down into sentences but no part of the text is discarded. We

[6] The mean number of paragraphs in a document was noted to be 26.7 and, the mean number of words per paragraph was noted to be 58.6.

wanted to match sentences between two documents to check if or not any meaningful match between sentences occur.

Although breaking a text down into sentences seem trivial, it is not so in case of legal case reports. Dots in a text might mean many things among which one is a sentence break. Although there are advanced ways of identifying sentence breaks, we decided to proceed through a simple intuitive technique. In our method, only those dots were considered to be a sentence break which met all of the following properties:

– It is not a dot among the list of common abbreviations such as *Mr., Mrs., Dr., Dt., aprox., Misc., No., vs., e.g., vis., P.T.O.,* etc.
– It has no other special characters within the vicinity of 5 alphabets.

After the detection of sentence breaks, the document is now partitioned into sentences, and the similarity calculation is now done as is done in case of paragraphs. Here, a document d is represented as a set of sentences $\{s_1, s_2, ..., s_n\}$, where d is the document and n is the total number of sentences in d. Given two documents d_1 and d_2 among which the similarity is to be computed, let d_1 have n_1 number of sentences and d_2 have n_2 number of sentences. Then, a total of $n_1 \times n_2$ similarity scores can be computed among the sentences of the two documents. To get the ultimate similarity value among the two case documents, we compute the *mean* of all the similarity scores.

4.1.6 RFCs of the document

Juridical case reports frequently incorporate references to earlier cases. The textual content encompassing such references are called *Reason for citation* (shortened as *RFC*), since these content bits demonstrate why a specific case report referred to another. Earlier works (Zhang and Koppaka 2007) guessed that such RFCs incorporate the significant juridical issues studied in a case report, which additionally give the reasons why a future case should refer to the current case.[7] Therefore, concentrating on the RFCs is by all accounts a promising method for recognizing the significant legal issues discussed in a case report. To construct the set of RFCs from a given report, we consider the 40 leading words and the 40 trailing words from where a reference was made in the text of the case report. Table 3 shows an illustration of RFCs from an example case report from our dataset of Indian Supreme Court case reports.

Utilizing RFCs to compute the relevance between two case reports is to some degree comparable to utilizing passages to represent a report (as discussed previously). When a document d has n number of references (to earlier cases), it can be expressed as a collection of RFCs $\{r_1, r_2, \ldots r_n\}$. Given the two documents d_1 and d_2 between which the similarity must be calculated, let n_1 and n_2 be the total number of

[7] LexisNexis, a well known legal search system (https://www.lexisnexis.com/), is known to be assisted by this principle.

Table 3 An extract from a legal case report indicating the RFCs. The references are underlined and, the RFCs are the italicised portions in the vicinity of the references.(The extract was taken from the legal case 2015_INSC_14 available at https://indiankanoon.org/doc/1934103/)

In Vellore Citizens' Welfare Forum v. Union of India and Others [(1996) 5 SCC 647] *the precautionary principles and polluter pays principle were held to be part of the environmental law of the country. It was held that the polluter pays principle means that the absolute liability for harm to the environment extends not only* to compensate the *victims of pollution but also the cost of restoring the environmental degradation. Remediation of the damaged environment is part of the process of sustainable development. In this very case, i.e., Research Foundation For Science Technology National Resource Policy v. Union of India & Anr.* [2003 (9) SCALE 303] *while examining the precautionary principle and polluter pays principle, the legal principles noticed in brief were :- "The legal position regarding applicability of the precautionary principle and polluter pays principle which are part of the concept of sustainable development in our country is now well settled. In Vellore Citizens' Welfare Forum v. Union of India & Ors.* [(1996) 5 SCC 647], *a three Judge Bench of this Court, after referring to the principles evolved in various international conferences and to the concept of "sustainable development", inter alia, held that the precautionary principle and polluter pays principle have now emerged and govern the law* in our country, as is clear from Articles 47, 48-A and 51- A (g) of our Constitution and that, in fact, in the various environmental statutes including the Environment (Protection Act, 1986, these concepts are already implied. These principles have been held to have become part of our law. Further, it was observed in *Vellore Citizens' Welfare Forum's case that these principles are accepted as part of the customary international law and hence there should be no difficulty in accepting them as part of our domestic law. Reference may also be made to the decision in Prof. M.V. Nayudu (Retd.) and Ors.* [(1996) 5 SCC 718] *where, after referring to the principles noticed in Vellore Citizens' Welfare Forum's Case, the same have been explained in more detail with a view to enable the Courts and the Tribunals or environmental authorities to properly apply the said principles in the matters which come before them.*

RFCs in d_1 and d_2 individually. Like the case with paragraphs, we process all conceivable $n_1 \times n_2$ similarity values between the RFCs of the two records, and afterwards, compute the *mean* of all these similarity values as the resultant similarity score between the two documents.

4.2 Measuring similarity among the document representations

After we have chosen the important parts (representations) from the textual content of the case reports (between which similarity is to be estimated), we now need to compute the similarities between the representations. As shown in Fig. 1, this is the second step of our experimental setup where any one of the seven embedding based similarity measurements is used.

The task of measuring similarity between two text snippets (representations) has earned widespread endeavour from the research communities of information retrieval and data mining, and a few popular systems already exist in the literature. The most famous systems utilize a vector representation of the two given texts, where the elements of the vector representation might be the words contained in the reports, or latent topics or even semantic ideas. When the vector representations of both the reports are acquired, the famous *cosine similarity* measure can be utilized to compute the similarity between the pair of vectors. Given two vectors A and B of dimensionality n each, the cosine similarity between the vectors is:

$$cos\_sim(A,B) = cos(\theta) = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}$$

where, A_i and B_i are the i^{th} coordinates of the vectors A and B individually, and, θ is the degree of rotation among A and B in the n-dimensional vector space.

The challenge is now concerning what is an efficient method for transforming the textual representation of a given case report to a set of numeric vector coordinates, with the goal that the meaning and context can be obtained precisely from these set of numbers. In the present work, we perform broad investigations upon six popular existing techniques and one self-devised strategy to compose the numeric vectors from a chosen text snippet, and afterwards compute the similarity between the vectors – (i) TF-IDF Vectorizer, (ii) Topic models, viz. Latent Dirichlet Allocation, (iii) Word embeddings, viz. Word2vec, and (iv) Document embeddings, viz. Doc2vec, (v) Law2Vec Embeddings, (vi) BERT Embeddings. In the remainder of this section, we depict how these six techniques enable us to form the vectors from any provided text snippet, how the related models were built and, how we use them to finally form the embedding(vector) for each snippet of texts.

4.2.1 Preprocessing the text

We apply some essential pre-processing and normalization operations over the textual segments, before the construction of vectors. These pre-processing methods help to shape better representations of any textual content.

1. Changing the case of all letters to small caps.
2. Tokenizing the textual content into constituent terms using pre-defined delimiters (double newlines, sentence-breaks)
3. Discard all words that contain any special characters (except from hyphen, underscore)
4. Filtering out standard English stopwords from the vocabulary of the text.
5. Stemming of all words, via the well known Porter Stemmer
6. Remove all words that appear in less than three case reports in the whole corpus. The aim of this is to coarsely clean most of the misspelled words and words that are exceptionally uncommon and probably won't add enough meaning to the representation of the case report.

4.2.2 TF-IDF Vectors

This is a basic yet capable methodology for transforming a given text into a set of numeric vector coordinates. A document d is expressed as a vector of a length same as the total number of non-redundant words in the dictionary of the corpus. The i^{th} component of the vector compares to the i^{th} word in the dictionary, which also, contains the TF-IDF score of the respective word, where the TF (term frequency) is computed as the number of occasions the word has appeared in d, and IDF computes the inverse of the number of occurrence of the word in the entire corpus. Although many variations of TF-IDF exists, we used the implementation given in the popular Python Machine Learning module *sklearn* (Pedregosa et al. 2011). In *sklearn*, the TF-IDF score of a single term with respect to a given text is calculated as

$$tfidf(t, d) = tf(t, d) \cdot idf(t)$$

where, d is the text that needs to be converted to a vector, and $tf(t, d)$ is the number of times t has occurred in d. The *idf* term is calculated during the training process as

$$idf(t) = \log \frac{n}{df(t) + 1}$$

where n is the total number of documents in the corpus.

To compute the similarity score between two text snippets, we calculate the cosine similarity between the two TF-IDF vectors respective to the two texts.

4.2.3 PScoreVect

PScore is a word scoring function just as is TF-IDF. The *PScoreVect* is very similar to the TF-IDF vectors, where instead of the TF-IDF scores we use *PScore* scores as described in Sect. 4.1.2. Here, the number of distinct terms (= n) in the dictionary of the entire collection of texts is considered as the dimension of the vector space. A document d, is represented by a vector of length n. The i^{th} element in the vector hence formed corresponds to the i^{th} word in the distinct set of corpus vocabulary. The i^{th} position in the vector has the numeric value equal to the *PScore* of the word. The exact mathematical representation of the PScore scoring function is described previously in Sect. 4.1.2 *PScore Summary*. For computing the similarity among two vectors, we simply find the cosine distance between the two *PScoreVect* vectors corresponding to the two documents.

4.2.4 Topic models, viz. latent dirichlet allocation (LDA)

Topic models are numerical models used to explore topics for a given corpus of text documents. For our experiments, the broadly accepted topic modeling algorithm Latent Dirichlet Allocation (LDA) (Blei et al. 2003) is used. LDA interprets topics as a combination of words (terms), and treats documents as a combination of topics.

The LDA is firstly trained over our entire corpus (collection of case reports). During the training of LDA, the desired number of topics has to be supplied as an input parameter, and the dimensionality of the vector provided by the LDA model is the same as the total number of topics. Although LDA models with higher dimension tend to perform better, they require a larger set of data to be trained upon. So, the ideal number of topics is generally restricted by the amount of training data. To determine the near optimal number of topics for our dataset, we performed experiments by varying it over five different values – 40, 80, 120, 160 and 200. It was found that the LDA model with 80 topics performed the best among all other variations. Hence, every method in our work that uses a LDA model, has its number of topic set to 80. Thus, for any given text snippet (e.g., a whole document, a paragraph, or a RFC), the LDA model gives a numeric vector (of size 80) which is expected to apprehend the topical distributions of the text. To compute the similarity across two representations, we calculate the cosine distance between the LDA vectors of the two representations.

4.2.5 Word embeddings – *Word2vec* and *Law2Vec*:

Word2vec (Mikolov et al. 2013) is a neural-based model that, when trained (in an unsupervised way) over a collection of texts, can construct a vector for each particular word in the collection. In case a large enough collection of text documents are available as examples for the model to generalize upon, the embeddings hence formed can hold important semantic information of the words. The embeddings of any two semantically comparable words (which have related contexts) will, in general, be situated closer to one another in the vector space.

To train the Word2Vec model we have used our dataset of Indian Supreme Court (INSC) cases containing over 33,500 case reports.[8] As input parameters, we need to provide the dimension of the numeric vector along with the type of model to be used (skip-gram or CBOW). Hence, we tried training Word2vec (Mikolov et al. 2013) with different sizes and with both type of models, it was observed that skip-gram models consistently outperformed CBOW models and the best among which had an embeddings size of 200. Hence, all results using Word2Vec presented in this paper use the skip-gram model with an embedding size of 200. All other hyperparameters were set to default, as set in Gensim.

In addition to the Word2Vec model trained using our own INSC dataset, we also make use of a pre-trained Word2Vec model on large legal corpus called *Law2Vec* (Ilias 2019). *Law2Vec* was trained upon a collection of 123,066 documents which was collected from eight different sources. The results are shown separately for both the Word2Vec models.

By design, Word2vec is built to provide us with embeddings for a given single word. But, for our experiments we wish to obtain vectors for a given text snippet (e.g., a paragraph, or an RFC) that contains multiple words. So, to obtain the resultant embedding for the text, it was needed to consolidate the embeddings of the comprising terms of a text segment. To this end, we calculate the *weighted mean of the vectors of the constituent terms of the text segment, where the vector of each term is weighted by the TF-IDF score of the term*. Note that Word2Vec embeddings are accepted as being additive and to have geometric properties such as compositionality(Corrêa Júnior et al. 2017), i.e., the vector acquired by combining the embeddings of two words is likely to encapsulate the broad context of the two words (Mikolov et al. 2013). Hence, taking a weighted mean of the Word2Vec embeddings should yield a good document embedding. This technique of combining word embeddings to obtain embeddings of larger texts such as paragraphs and

[8] We have used the Word2vec implementation from Gensim, an open-source package available at (https://radimrehurek.com/gensim/models/word2vec.html).

sentences has been used in many previous works (Corrêa Júnior et al. 2017; Iacobacci et al. 2016; Belinkov et al. 2015).

Once we get the Word2vec embeddings for two text snippets, the similarity between the text snippets is simply calculated as the cosine distance between the two embeddings.

4.2.6 Document embeddings viz. *Doc2vec*

This is an embedding technique devised by Le and Mikolov (2014) and is an extension to the model used in *Word2vec*. It is comprehensible from the name of the model that unlike *Word2Vec*, this model directly transforms a given text snippet (a document) into a vector. We do not need to combine the individual word vectors to form the text embedding in this case. According to Le and Mikolov (2014), not only does this model consider the words along with its surrounding text but also the sequence in which individual word appears in the text. Just as in case of Word2vec, we have used our whole corpus of INSC dataset (Indian Supreme Court Dataset) to train a Doc2vec model.[9]

Doc2vec requires the size of the embeddings to be given as a parameter value. Although a Doc2Vec model with higher number of dimensions tends to perform better than that with lower number of dimensions, the performance tends to saturate once the number of dimensions reaches a threshold value. In their original paper (Le and Mikolov 2014), this saturation was observed at around 400 dimensions. But a model with higher dimension requires a larger dataset to be trained upon. So, this number is dependent on the amount of data we have to train our model. To determine the dimension size that best suits our task, we measure its performance using various embedding sizes and find that Doc2vec embeddings with a size of 100 perform superior compared to the rest. Hereinafter, every result for Doc2vec presented in this paper employs a model with a size of 100 dimensions. We used the PV-DM variation of the model as implemented in Gensim with all other hyperparameters set to defaults.

The calculation of the similarity scores is straightforward when we are given the Doc2vec embeddings of the pair of text snippets – we essentially calculate the cosine distance between these two vectors.

4.2.7 BERT embeddings

BERT or Bidirectional Encoder Representations from Transformers (Devlin et al. 2018) is a language modeling technique and is a way of pre-training a neural model where the pre-trained model can be readily used for a variety of NLP tasks with an optional fine-tuning step in between. We have tried both – (1) *training the model from scratch*, and (2) *fine-tuning the pre-trained model*. To train the model we used the same dataset of 33, 500 Indian Supreme Court Case Reports as was also used for

[9] We used the Doc2vec implementation from the Gensim, an open-source package available at (https://radimrehurek.com/gensim/models/doc2vec.html).

training all the other models. In case of fine-tuning we have used the BERT uncased model[10] and hence used it to convert sequences of text into a fixed length of numeric vector as is done by Doc2Vec (Sect. 4.2.6) or LDA (Sect. 4.2.4). In case of training from scratch we have used the BERT's 12 layer architecture with 768 hidden layers and 12 attention heads.

It was found that training the model from scratch actually gave better results. This might be due to the fact that the legal text is considerably different in structure from other generic forms of literary texts. The process of fine-tuning not only had to incorporate a large number of new words (especially legal terms) in the vocabulary but also had to learn the associations between these new terms.

Although BERT can readily convert a given text into a vector, one major drawback is that it has a limit for the length of the input it accepts. It can not accept strings with more than 512 words. This is because the self-attention layer in BERT has a quadratic complexity $O(n^2)$ with respect to the length n of the input sequence. Although alternative methods (Ainslie et al. 2020; Beltagy et al. 2020; Pappagari et al. 2019) exist to alleviate this limitation but, we experimented with a simpler adaptation that does not modify the architecture of the original BERT model. A simple workaround is to break the text into overlapping chunks and find the embeddings for each chunk and then combine these embeddings to form the final embedding. To form the chunks, we consider contiguous sequences of 500 words with an overlap of 50 words between two consecutive text chunks. We use the BERT model (trained as described above) to obtain an embedding for each chunk, and then take an element-wise mean of these individual chunk-embeddings to get the final embedding for the entire text snippet. Thus, when given two text snippets, we convert the snippets into two fixed-length numeric vectors and simply find the cosine distance between these two vectors, which is considered to be the final similarity score between the two given text snippets.

4.3 Summary

In this section, we discussed different choices for two key steps that are used in our experimentations for estimating the similarity between two legal case reports-(1) how to represent a document text, and (2) how to compute the similarity between the text representations of the two documents. This fact is illustrated briefly in Fig. 1.

We have eight methods of representing the document text. These can be classified into two major categories:

- Consolidated representation: Full Document, MyScore Summary, PScore Summary, PS catches and CRF catches.
- Fragmented representation: RFCs, Sentences and Paragraphs.

[10] The pre-trained BERT model is available at https://storage.googleapis.com/bert_models/2018_10_18/uncased_L-12_H-768_A-12.zip.

The similarity measurement of documents using fragmented representations are a little different. Unlike in case of Consolidated representations, Fragmented representations provide us with multiple parts of text snippets from both the text documents among which the similarity is to be measured. Let us suppose, that document d_1 has n_1 number of fragmented texts, and document d_2 has n_2 number of fragments. Now, it is possible to calculate a total of $n_1 \times n_2$ number of similarities among these fragments. In our experiments we have tried to consider the final similarity value as either the maximum of these $n_1 \times n_2$ similarities, or have tried using the average of them. We observed that taking the average of these values showed better performances. So, for the fragmented representations of text, we have performed the similarity measurements in the above manner.

For the second step, we examine seven distinct methods of measuring the similarity score – TF-IDF, topic models (LDA), word embeddings (Word2Vec), pre-trained word embeddings on large legal corpus (Law2Vec), document embeddings (Doc2Vec), and a state-of-the-art language model (BERT). To select our method of document similarity measurement, we can choose any document representation and vectorize it using any vectorizing method. In the next section, we finally present the comparison across the different representations and techniques for measuring similarity between two given documents.

5 Experimental setup and results

In this section, we describe various experiments to investigate (1) which are the best representations for a given legal document, and (2) which are the most potent document similarity techniques. For a detailed evaluation of all the methods, we make a two-fold comparison between the obtained similarity scores with those provided by the domain experts. Described below are the experimental setup along with the final evaluation of the similarity measures.

5.1 Baseline method

The baseline method considered here is a technique proposed by (Kumar et al. 2013). The authors of this paper have utilized a technique they call *Paragraph Link*, where the objective is to discover how comparable the paragraphs of two reports are. They apply a combined strategy made out of text-based and network-based methodologies. A brief discussion on the baseline method is as follows.

The initial step is to divide the document text into constituent paragraphs. Secondly, by using the TF-IDF scores of every word present in a paragraph, a paragraph vector is formed for every paragraph in each document. Finally, a paragraph-link is attached between the pair of paragraphs if the cosine distance between the individual paragraph vectors is greater than a threshold value of 0.3 (the value utilized by (Kumar et al. 2013)). Hence, when presented with two case reports, we check the total count of shared paragraph-links between the two documents. In the event that the count of shared paragraph-links between two reports is equal to or higher than three, they are said to be similar. The number of shared paragraph-links between the two reports is said to be the final similarity score.

It is apparent that the dataset utilized by the work of (Kumar et al. 2013) was made out of 3,866 Indian Supreme Court cases. Although for our situation, we have utilized a bigger dataset of 33,545 cases, which is an order of magnitude bigger. This dataset was utilized to calculate the number of shared paragraph-links. Henceforth, the total count of paragraph-links for a report in our dataset is comparatively higher than that in the original paper, which prompts a lot higher scores in our final results – Table 6 – when contrasted with the quantity of PLs referenced by (Kumar et al. 2013).

5.2 Evaluation metrics

To assess each of our strategies we repeat the following for every method. First, we measure similarity scores for 47 pairs of test documents by the given method. For a given pair of documents, the similarity score computed by a method is simply the cosine similarity between the vectors corresponding to the two documents. Once the similarity scores for every pair of documents is obtained, we then compare them with similarity scores given by experts. This comparison among the obtained similarity scores and the expert similarity scores is done using two well-known metrics: (i) Pearson Correlation Coefficient, and (ii) Accuracy.

5.2.1 Pearson correlation coefficient

Pearson correlation coefficient is employed to decide how well our strategies perform when contrasted with the similarity values given by the professionals. For any two variables, the Pearson Correlation Coefficient is simply covarinace divided by its standard deviation. Given two variables X and Y, Pearson correlation coefficient (ρ) is defined as:

$$\rho = \frac{cov(X, Y)}{\sigma_X.\sigma_Y} \tag{5}$$

where σ_X and σ_Y are the standard deviations of the two variables and $cov(X, Y)$ is the covariance of the variables. The range of correlation coefficient(ρ) lies within $[-1.0, 1.0]$. A correlation score of 1.0 suggests that the variables are perfectly correlated. In our evaluation, to compute the correlation score we have the two variables as the expert similarity score and, the method-yielded similarity value (obtained by a particular strategy among the 56 different strategies discussed in this work). Each observation in a variable is a similarity score between a pair of given case reports.

5.2.2 Accuracy

Our problem of measuring document similarity can be remodeled as a problem of identifying similar document-pairs, essentially converting it into a two-class classification problem where one class is *'similar'* and the other is *'not similar'*. Given

Table 4 Pearson Correlation Coefficients for all methods of measuring document similarity (in each column) with respect to the type of text representation employed (in each row). The best performing score in each row is in boldface and the topmost score in every column is in italics. The highest three Correlation values in the whole table are underlined

Representations / Methods	W2V	L2V	D2V	TFIDF	LDA	PScoreVect	BERT
Full doc	*0.595*	*0.623*	***<u>0.685</u>***	<u>0.617</u>	0.583	*0.327*	0.271
MyScore summary	0.561	0.430	0.486	0.590	***0.612***	0.221	0.280
Pscore summary	0.367	0.521	**0.538**	0.530	0.412	0.168	0.218
CRF catches	0.346	0.395	0.023	***<u>0.654</u>***	0.571	0.231	0.278
PS catches	**0.512**	0.420	-0.256	0.408	0.298	0.096	0.055
RFCs	0.375	0.300	0.324	0.370	**0.508**	-0.061	0.129
Sentences	0.416	0.230	0.243	0.486	**0.563**	-0.130	0.174
Paragraphs	0.584	0.506	**0.586**	0.512	0.564	0.025	0.295
Mean correlations	0.470	0.428	0.329	**0.521**	0.514	0.110	0.213

a method of similarity measurement, a document-pair is considered to be *'predicted similar'* if the similarity score is higher than 0.5, otherwise it is considered to be *'predicted not similar'* (note that for every method of similarity measurement discussed in this work, the similarity score obtained lies in the range [0, 1]). If, a pair has an expert score more than 5, it is considered to be *'actually similar'*, otherwise considered to be *'actually not similar'* (note that for every pair of documents the expert-scores lie within the range of [0, 10]). For a given pair of documents along with its expert similarity score and the calculated similarity score, we can have four situations:

1. True positive (TP): it is actually similar and also predicted similar
2. True negative (TN): it is actually not similar and also predicted not similar
3. False positive (FP): it is actually not similar but predicted similar
4. False negative (FN): it is actually similar but predicted not similar

For a particular method of similarity measurement, there are a total of 47 similarity scores (for the 47 document-pairs), each of which falls under any one of the four categories just mentioned. Hence, for every method, $TP + TN + FP + FN = 47$. Accuracy in terms of these four variables is defined as $Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$ or simply in our case, $Accuracy = \frac{TP+TN}{47}$.

5.3 Results

In this section, the varied results obtained after our experimentations are discussed. We first investigate which of the methods discussed in this work yield the most successful similarity measures. To do this, the predicted similarity scores of each method is compared with the expert similarity scores using the two chosen

Table 5 Accuracy Measures for all methods of measuring document similarity (in each column) with respect to the type of text representation employed (in each row). The best performing score in each row is in boldface and the topmost score in every column is in italics. The highest three Correlation values in the whole table are underlined

Representations / Methods	W2V	L2V	D2V	TFIDF	LDA	PScoreVect	BERT
Full doc	0.766	0.723	0.768	<u>***0.872***</u>	0.723	*0.723*	0.660
MyScore summary	0.660	0.702	0.745	<u>***0.872***</u>	0.787	0.673	0.702
Pscore summary	0.723	*0.787*	0.714	<u>**0.857**</u>	*0.837*	0.673	0.702
CRF catches	0.702	0.702	0.660	**0.830**	0.766	0.694	0.681
PS catches	0.766	0.723	0.660	0.681	**0.745**	0.673	0.660
RFCs	0.702	0.596	0.681	0.681	**0.766**	0.660	0.681
Sentences	0.766	0.638	0.596	**0.837**	0.796	0.612	*0.723*
Paragraphs	*0.787*	0.660	*0.787*	**0.830**	0.745	0.660	0.660
Mean accuracy	0.734	0.691	0.701	**0.808**	0.771	0.671	0.684

metrics *Correlation* and *Accuracy*. The metrics of comparison are elucidated in Sect. 5.2. The correlation and accuracy values are presented in Table 4 and Table 5 respectively.

Later in this section, we select the top five best-performing methods and present the exact similarity scores given by these methods for all 47 pairs of documents in Table 6.

5.3.1 Comparison among the different document representations and different similarity measures

Given the various ways to represent a legal document and several methods to compute similarity among them, we presently seek to determine the most suitable ones. Table 4 and Table 5 shows the performance of seven different similarity measuring techniques – (*Word2Vec, Law2Vec, Doc2Vec, TFIDF, LDA, PscoreVect, and BERT*) each of which are applied upon eight different document representations – (*Whole Document*, *MyScore Summary*, *PScore Summary*, *CRF Catches*, *PSlegal Catches*, *RFC*, *Sentences* and *Paragraphs*). All values shown in Table 4 are Pearson Correlation Coefficient values, and all values shown in Table 5 are Accuracy scores. Hence, we have experimented with, and presented results for seven different methods of similarity measurement with respect to eight different text representations ($7 \times 8 = 56$ experiments in total) using two metrics of evaluation.

Among all the methods of similarity measurement reviewed in this work, the method whose similarity scores were found to correlate the most with the expert similarity scores was the Doc2Vec similarity over the whole documents. This observation is somewhat unexpected – from our discussions with law experts, we felt that different parts/paragraphs of a case report in general examine different legal issues; consequently, we had expected paragraph-based techniques to produce better results. It can be noted that, the gold standard considered in this work was created by legal specialists inspecting two entire document texts and assessing their similarity. This

Table 6 Similarity measures for 47 pairs of document, as inferred by (i) legal experts, (ii) the baseline methodology (Kumar et al. 2013), and (iii) the best five methodologies discussed in this work The scores inferred by our methods have much higher correlation and accuracy, as compared to the baseline methodology. The best values of correlation and accuracy are marked with boldface

Sl.	Case pairs	Expert scores	Baseline scores	Whole doc with doc2vec	CRF catches with TFIDF	Full doc with TFIDF	MyScore summary with TFIDF	PScore summary with TFIDF
1	1992_47 & 1992_76	0	78	0.160	0.030	0.109	0.044	0.054
2	1992_76 & 1992_182	0	49	0.146	0.310	0.091	0.040	0.069
3	1972_11 & 1984_115	0	745	0.084	0.325	0.091	0.065	0.055
4	1969_57 & 1980_91	0	2303	0.271	0.099	0.259	0.120	0.244
5	1959_151 & 1982_28	0	522	0.238	0.080	0.110	0.021	0.081
6	1976_200 & 1959_151	0	583	0.051	0.250	0.138	0.015	0.110
7	1985_114 & 1959_151	0	756	0.263	0.173	0.138	0.028	0.106
8	1966_236 & 1967_267	0	322	0.353	0.075	0.232	0.078	0.133
9	1961_34 & 1979_110	0	486	0.322	0.148	0.344	0.088	0.204
10	1961_34 & 1987_37	0	146	0.193	0.000	0.084	0.057	0.052
11	1992_47 & 1987_315	0	279	0.358	0.148	0.416	0.070	0.486
12	1971_138 & 1992_47	0	645	0.459	0.157	0.549	0.334	0.348
13	1992_47 & 1992_76	0	78	0.160	0.000	0.109	0.044	0.054
14	1984_115 & 1987_315	0	390	0.238	0.446	0.186	0.100	0.148
15	1983_129 & 1983_27	1	806	0.561	0.164	0.583	0.287	0.610
16	1979_110 & 1953_28	2	327	0.178	0.172	0.433	0.233	0.448
17	1963_170 & 1979_158	2	2006	0.492	0.462	0.333	0.231	0.293
18	1983_27 & 1983_37	2	808	0.527	0.363	0.463	0.376	0.496
19	1983_27 & 1979_33	2	2420	0.581	0.205	0.415	0.383	0.472
20	1984_115 & 1981_49	2	536	0.500	0.220	0.533	0.178	0.419
21	1979_110 & 1989_233	3	356	0.351	0.359	0.303	0.109	0.534
22	1983_129 & 1976_176	5	574	0.266	0.442	0.402	0.150	0.368
23	1971_111 & 1972_291	5	1098	0.393	0.389	0.551	0.107	0.566

Table 6 (continued)

Sl.	Case pairs	Expert scores	Baseline scores	Whole doc with doc2vec	CRF catches with TFIDF	Full doc with TFIDF	MyScore summary with TFIDF	PScore summary with TFIDF
24	1990_171 & 1988_88	5	2058	0.297	0.646	0.221	0.280	0.133
25	1972_31 & 1984_115	5	2429	0.536	0.335	0.507	0.257	0.392
26	1984_118 & 1971_336	5	968	0.356	0.536	0.290	0.049	0.205
27	1987_154 & 1964_144	5	839	0.492	0.526	0.709	0.737	0.563
28	1973_186 & 1986_218	5	957	0.393	0.135	0.312	0.213	0.327
29	1990_96 & 1990_171	5	692	0.439	0.460	0.393	0.216	0.186
30	1958_3 & 1992_144	5	1073	0.372	0.396	0.458	0.400	0.459
31	1979_158 & 1965_111	7	618	0.529	0.341	0.352	0.610	0.660
32	1962_303 & 1972_291	7	535	0.540	0.594	0.645	0.606	0.641
33	1987_37 & 1989_233	7	32	0.234	0.027	0.074	0.017	0.080
34	1953_40 & 1953_24	7	2526	0.836	0.916	0.904	0.755	0.904
35	1966_154 & 1976_43	7	3223	0.431	0.249	0.384	0.635	0.605
36	1953_24 & 1957_52	7	3532	0.177	0.140	0.298	0.136	0.259
37	1984_115 & 1971_49	7	721	0.482	0.550	0.597	0.607	0.579
38	1980_221 & 1984_115	8	934	0.539	0.670	0.405	0.646	0.233
39	1980_39 & 1969_324	8	538	0.648	0.370	0.916	0.731	0.955
40	1991_48 & 1987_189	9	491	0.537	0.880	0.724	0.164	0.718
41	1979_104 & 1979_110	9	661	0.695	0.353	0.793	0.380	0.745
42	1985_113 & 1969_324	9	674	0.619	0.569	0.906	0.925	0.987
43	1979_33 & 1979_110	9	701	0.838	0.362	0.885	0.636	0.782
44	1968_197 & 1972_62	10	4265	0.584	0.675	0.608	0.644	0.219
45	1992_47 & 1984_115	10	487	0.540	0.590	0.559	0.307	0.416
46	1991_12 & 1985_113	10	675	0.725	0.797	0.973	0.847	0.973

Table 6 (continued)

Sl.	Case pairs	Expert scores	Baseline scores	Whole doc with doc2vec	CRF catches with TFIDF	Full doc with TFIDF	MyScore summary with TFIDF	PScore summary with TFIDF
47	1983_37 & 1979_33	10	2368	0.750	0.643	0.844	0.636	0.880
Correlation			0.333	**0.685**	0.617	0.654	0.590	0.530
Accuracy			0.489	0.768	0.830	**0.872**	**0.872**	0.857

process of expert annotation may be a reason why estimating semantic similarity between the two full document texts relates best with the similarity scores specified by the legal specialists.

Although Doc2Vec performs markedly well with full documents, paragraphs and summaries, it does not perform well enough with the other form of text representations. This might be due to the fact that linguistic structure in the input text is needed to use Doc2Vec as a good embedding technique, as is provided during it's training. As shown in the bottom row of Table 4, it has the third lowest correlation value when averaged over all document representations and compared with other similarity measurement techniques. It also has the fourth lowest score when the accuracy scores are averaged over all document representations, as shown in the bottom row of Table 5. So, other forms of text representations need to be explored.

Apart from using Doc2vec embeddings of full documents, high correlation is also evident in two other methods: – TFIDF similarity using Full documents and, TFIDF similarity using CRF catches. In this regard it is to be mentioned that, CRF catches are very low in number for any given document. On an average there were just 10.6 CRF catches per document. This low number of CRF catches yields a faster mechanism of similarity measurement and might be useful in cases of practical implementation where, a system might need to find similar prior judgements as results when given the current case description as query.

The performance of BERT is unexpectedly low in our setup, for which there are two potential reasons. First, in its innate form, BERT does not perform well with long sequence of texts. Since legal documents are long, some technique needs to be adopted to obtain a representation/embedding of a document using BERT. To this end, we obtain the document-embeddings by combining embeddings of overlapping chunks (as stated in Sect. 4.2.7). It appears that this methodology does not work well; we leave it as future work to identify better ways of obtaining transformer-based embeddings of long legal documents. Second, the BERT model consists of a eight-layer deep transformer architecture consisting of 108M parameters to be trained. It is understandable that a dataset of 33, 545 documents is largely insufficient for training such a large model.

Apart from correlation, accuracy is another important metric of evaluation (accuracy values are reported in Table 5). Methods of similarity measurement that provide the highest accuracy values are: – TFIDF similarity between full doc, TFIDF similarity between MyScore Summaries and, TFIDF similarity between PScore Summaries. It shows that Summaries are a good way of representing documents while measuring similarity between them. These summary representations are not originally proposed in any previous work. These representations are adapted in this work from two popular word scoring functions: – *PScore* by (Mandal et al. 2017c) and, *MyScore* by (Galgani et al. 2012) as described in Sect. 4.1.2.

All these results can be summarised as:

- Traditional vectorisers (such as TF-IDF and LDA) perform well in case of document similarity.
- Doc2Vec performs well for document representations where linguistic structure is present such as summaries, paragraphs and full document texts.

- BERT embeddings do not perform well for the task, probably due to less amount of training data and long input texts.
- The top three methods by Correlation Coefficient are:
  - Doc2Vec between Full Documents
  - TF-IDF using Full Documents
  - TF-IDF using CRF Catches
- The top three methods by Accuracy are:
  - TF-IDF using Full Documents
  - TF-IDF using MyScore Summaries
  - TF-IDF using PScore Summaries

5.3.2 Comparison with the baseline

In this section, the top five methods of measuring document similarity are contrasted with the similarity scores provided by the law specialists. To choose the top five methods, we inspect the Tables 4 and 5. As mentioned earlier, the best values in each table are the underlined ones. Taking a distinct set of the methods corresponding to the underlined scores in two tables, gives us five methods as follows:

1. *Doc2Vec* similarity between *Full Docs*
2. *TFIDF* similarity between *CRF Catches* †
3. *TFIDF* similarity between *Full Docs*,
4. *TFIDF* similarity between *MyScore Summaries* †
5. *TFIDF* similarity between *PScore Summaries* †

Among the five methods in the above list, the three methods marked with a † sign are our own devised methods, and the rest two are pre-existing methods. Finally, a comparison is made between the above five top performing methodologies with the baseline method (Kumar et al. 2013). We do a two-fold comparison of these five methods, based on their correlation and accuracy scores with the expert judgements.

Table 6 shows the seven different similarity scores – as computed by (1) the legal experts, (2) the baseline method, and (3) *D2V* similarity between *Full Documents*, (4) *TFIDF* similarity between *Full Documents*, (5) *TFIDF* similarity between *Full Docs*, (6) *TFIDF* similarity between *MyScore Summaries*, (7) *TFIDF* similarity between *PScore Summaries* – for all 47 document-pairs (for which the expert judgement was obtained from the baseline paper by (Kumar et al. 2013)). In the second to last row in the table, the Pearson correlation coefficient for each technique w.r.t. the scores assigned by experts is given. In the last row, the Accuracy measure of the methods are given. Note that the correlation values lie in the range of $[-1.0, 1.0]$ and those of accuracy scores lie in the range of $[0.0, 1.0]$. The higher the value of the correlation and accuracy, the superior is the performance of the specific method.

As found in the second to last row of the table, the correlation value 0.33 for the baseline technique is markedly lower than the correlation value 0.685, 0.617, 0.654, 0.590 and, 0.530 produced by the proposed methods of measuring document similarity. In the last row of the table, the accuracy measure for the baseline is 0.489, which is again remarkably lower than the accuracy measures of the proposed five methods which provide us with scores as high as 0.768, 0.830, 0.872, 0.872 and, 0.857. Hence, the five proposed methods of measuring document similarity produces far superior results over the baseline.

6 Concluding discussions

In this paper, we experimented with several methodologies for quantifying the similarity between two court case documents and finally recommend five of our best-performing methods for computing similarity among case reports. Among the best performing five methods (as described in Sect. 5.3.2), three were our original work and the rest two are pre-existing information retrieval methods adapted for the specific task of legal information retrieval. Previously, the text-based systems that were utilized for this objective were constrained to elemental frequency-based methods. As far as we believe, this is the first attempt at using advanced semantic techniques for instance, topic modeling and neural network-based methodologies (document embeddings and word embeddings) for measuring similarity between legal documents. We show that the embedding based and text-based similarity measures, presented in this paper, significantly out-perform the baseline technique that utilize both text-based (TF-IDF) and network-based similarity measures (Kumar et al. 2013).

From our experiments, it is observed that neural-network based models do not perform too well in case of legal documents due to the low volume of texts available to be trained upon. Traditional methods perform comparably well along with the neural-network based embedding techniques. In specific, the vectorizing methods that use bag-of-words (BOW) representation of text such as the TF-IDF and LDA perform better than those that use localized contextual information such as the Word2Vec and BERT (probably due to the relatively small amount of training data). Nevertheless document-level contextual information is helpful as observed in the case of Doc2Vec. In future, it might prove worthy to build novel models that capture document-level contextual information.

Note that, the effectiveness of the similarity measures that use summaries as document representations, depends profoundly on the nature of the summaries being produced. Better summarization would inherently bring about better similarity scores. Along these lines, the effectiveness of the summary based measures can potentially be improved in future by developing better automatic summarization systems - we intend to proceed in this way in future. Then again, techniques like topic modeling and Doc2vec can be promptly applied over the entire report text, or over passages of a report.

It is noteworthy, that among all the similarity measuring techniques described in this work, some are more easily applicable than others. For example, methods that employ RFCs as document representations, can only be practical when the citation points are identified in a text. Hence, this methods will not be useful when the legal practitioner has a brief sketch of the ongoing case.

Among other considerations in a system of precedent retrieval, we have studied and experimentally decided upon how well similarity can be measured between two given case reports. In certain scenarios, there might be other important issues to be considered such as – whether or not the method is scalable for large datasets, whether or not the method has a good response time to be applicable for a real-time retrieval system, whether the system can capture the user-query via only a few keywords instead of a description, whether the user is trying to search for only some issue within the case and not the complete case report, and so on. These considerations are brought about when a practical system is to be implemented. Our next endeavour is to build a complete and robust precedent retrieval system, utilizing the insights obtained from the present study.

Additionally, we look to improve the methodology of similarity computation in future, by designing some machine-learning based customized representation of court case documents to measure similarity across documents. For instance, a promising direction is to first segment a case document into various rhetorical segments (Bhattacharya et al. 2019) and then measure similarity between each segment. Again, state-of-the-art embedding techniques meant for long documents (Ainslie et al. 2020; Beltagy et al. 2020; Pappagari et al. 2019) can be tried for generating better representations of long legal documents. Also, transformer-based similarity matching models such as Sentence-BERT (Reimers and Gurevych 2019) can be applied for the task, after suitable variations are developed for working with long documents. Finally, though text-based methods presented in this work perform well, they can be further improved by incorporating information from a well-connected legal citation graph. In future, we hope to design advanced techniques of similarity measurement by hybridizing embedding-based measures with citation graph information.

Funding The first author is supported by the Visvesvaraya PhD scheme from the Ministry of Electronics and Information Technology (Grant No. VISPHDMEITY-1570).

References

Ahmad WU, Bai X, Peng N, Chang K (2018) Learning robust, transferable sentence representations for text classification. CoRR abs/1810.00681, arXiv1810.00681

Ainslie J, Ontanon S, Alberti C, Cvicek V, Fisher Z, Pham P, Ravula A, Sanghai S, Wang Q, Yang L (2020) Etc: Encoding long and structured inputs in transformers. arXiv2004.08483

Backstrom L, Kleinberg J (2014) Romantic partnerships and the dispersion of social ties: A network analysis of relationship status on facebook. In: Proc. ACM Conference on Computer Supported Cooperative Work & Social Computing (CSCW), pp 831–841

Batet M, Sánchez D, Valls A, Gibert K (2013) Semantic similarity estimation from multiple ontologies. Applied intelligence 38(1):29–44

Belinkov Y, Mohtarami M, Cyphers S, Glass J (2015) VectorSLU: A continuous word vector approach to answer selection in community question answering systems. In: Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)

Beltagy I, Peters ME, Cohan A (2020) Longformer: The long-document transformer. arXiv2004.05150

Bhattacharya P, Paul S, Ghosh K, Ghosh S, Wyner A (2019) Identification of rhetorical roles of sentences in indian legal judgments. In: Proceedings of International Conference on Legal Knowledge and Information Systems (JURIX)

Blei DM, Ng AY, Jordan MI (2003) Latent dirichlet allocation. J Mach Learn Res 3:993–1022

Brüninghaus S, Ashley KD (2001) Improving the representation of legal case texts with information extraction methods. In: Proceedings of the International Conference on Artificial Intelligence and Law, ICAIL '01, pp 42–51

Chen Q, Peng Y, Lu Z (2018) Biosentvec: creating sentence embeddings for biomedical texts. CoRR abs/1810.09302, http://arxiv.org/abs/1810.09302, 1810.09302

Corrêa Júnior EA, Marinho VQ, dos Santos LB (2017) NILC-USP at SemEval-2017 task 4: a multi-view ensemble for twitter sentiment analysis. In: Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017), pp 611–615

Devlin J, Chang MW, Lee K, Toutanova K (2018) Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:181004805

Ekbal A, Haque R, Bandyopadhyay S (2008) Named entity recognition in Bengali: a conditional random field approach. In: Proceedings of the International Joint Conference on Natural Language Processing: Volume-II

Galgani F, Compton P, Hoffmann A (2012) Towards automatic generation of catchphrases for legal case reports. Springer, Berlin Heidelberg, pp 414–425

Iacobacci I, Pilehvar MT, Navigli R (2016) Embeddings for word sense disambiguation: an evaluation study. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp 897–907

Ilias C (2019) Law2Vec - Legal Word Embeddings by Ilias Chalkidis. https://archive.org/details/Law2Vec

Kumar S, Reddy PK, Reddy VB, Singh A (2011) Similarity analysis of legal judgments. In: Proc. ACM Compute Conference, pp 17:1–17:4

Kumar S, Reddy PK, Reddy VB, Suri M (2013) Finding similar legal judgements under common law system. Springer, Berlin, pp 103–116

Le Q, Mikolov T (2014) Distributed representations of sentences and documents. In: Jebara T, Xing EP (eds) Proc. International Conference on Machine Learning (ICML), JMLR Workshop and Conference Proceedings, pp 1188–1196

Liu B, Niu D, Wei H, Lin J, He Y, Lai K, Xu Y (2019) Matching article pairs with graphical decomposition and convolutions. In: Proceedings of the Conference of the Association for Computational Linguistics (ACL), pp 6284–6294

Luo F, Xiao H, Chang W (2011) Product named entity recognition using conditional random fields. In: 2011 Fourth international conference on business intelligence and financial engineering, IEEE, pp 86–89

Mandal A, Chaki R, Saha S, Ghosh K, Pal A, Ghosh S (2017a) Measuring similarity among legal court case documents. In: Proceedings of the 10th Annual ACM India Compute Conference, Association for Computing Machinery, Compute '17, p 1–9, https://doi.org/10.1145/3140107.3140119

Mandal A, Ghosh K, Bhattacharya A, Pal A, Ghosh S (2017b) Overview of the FIRE 2017 irled track: information retrieval from legal documents. In: Working notes of FIRE 2017 - Forum for Information Retrieval Evaluation, pp 63–68

Mandal A, Ghosh K, Pal A, Ghosh S (2017c) Automatic catchphrase identification from legal court case documents. In: Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, ACM, New York, NY, USA, CIKM '17, pp 2187–2190, http://doi.acm.org/10.1145/3132847.3133102

Mikolov T, Chen K, Corrado G, Dean J (2013) Efficient estimation of word representations in vector space. CoRR abs/1301.3781

Minocha A, Singh N, Srivastava A (2015) Finding relevant indian judgments using dispersion of citation network. In: Proc. International Conference on World Wide Web (WWW) Companion, pp 1085–1088

Pappagari R, Żelasko P, Villalba J, Carmiel Y, Dehak N (2019) Hierarchical transformers for long document classification. 1910.10781

Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, Vanderplas J, Passos A, Cournapeau D, Brucher M, Perrot M, Duchesnay E (2011) Scikit-learn: machine learning in python. J Machine Learn Res 12:2825–2830

Peters ME, Neumann M, Iyyer M, Gardner M, Clark C, Lee K, Zettlemoyer L (2018) Deep contextualized word representations. CoRR abs/1802.05365, 1802.05365

Pvs A, Karthik G (2006) Part-of-speech tagging and chunking using conditional random fields and transformation based learningPVS. Shallow parsing for south asian languages 21:21–24

Reimers N, Gurevych I (2019) Sentence-bert: Sentence embeddings using siamese bert-networks. In: Proc. Conference on Empirical Methods in Natural Language Processing (EMNLP), pp 3982–3992

Santos E, Santos EE, Nguyen H, Pan L, Korah J (2011) A large-scale distributed framework for information retrieval in large dynamic search spaces. Appl Intell 35(3):375–398

Silfverberg M, Ruokolainen T, Lindén K, Kurimo M (2014) Part-of-speech tagging using conditional random fields: Exploiting sub-label dependencies for improved accuracy. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), Association for Computational Linguistics, Baltimore, Maryland, pp 259–264

Sugathadasa K, Ayesha B, de Silva N, Perera AS, Jayawardana V, Lakmal D, Perera M (2018) Legal document retrieval using document vector embeddings and deep learning. In: Science and Information Conference, Springer, pp 160–175

Toutanova K, Klein D, Manning CD, Singer Y (2003) Feature-rich part-of-speech tagging with a cyclic dependency network. In: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1, Association for computational Linguistics, pp 173–180

Tran V, Le Nguyen M, Tojo S, Satoh K (2020) Encoded summarization: summarizing documents into continuous vector space for legal case retrieval. Artificial Intelligence and Law pp 1–27

Zhang P, Koppaka L (2007) Semantics-based legal citation network. In: Proceedings of the 11th International Conference on Artificial Intelligence and Law (ICAIL), pp 123–130

Affiliations

Arpan Mandal[1] · **Kripabandhu Ghosh**[2] · **Saptarshi Ghosh**[3] · **Sekhar Mandal**[1]

Kripabandhu Ghosh
kripa.ghosh@gmail.com

Saptarshi Ghosh
saptarshi@cse.iitkgp.ac.in

Sekhar Mandal
sekhar@cs.iiests.ac.in

1 Department of Computer Science and Technology, Indian Institute of Engineering Science and Technology, Howrah, Shibpur, India

2 Department of Computational and Data Sciences (CDS), Indian Institutes of Science Education and Research, Kolkata, West Bengal, India

3 Department of Computer Science and Engineering, Indian Institute of Technology Kharagpur, Kharagpur, West Bengal, India