Overview of the FIRE 2019 AILA Track: Artificial Intelligence for Legal Assistance

Paheli Bhattacharya[1], Kripabandhu Ghosh[2], Saptarshi Ghosh[1], Arindam Pal[3], Parth Mehta[4], Arnab Bhattacharya[5], and Prasenjit Majumder[4]

[1] Indian Institute of Technology Kharagpur
[2] Tata Research Development and Design Centre (TRDDC), Pune
[3] Data61, CSIRO, Sydney, Australia
[4] DA-IICT Gandhinagar
[5] Indian Institute of Technology Kanpur

Abstract. The FIRE 2019 AILA track focused on creating a framework for evaluating different methods of retrieving relevant prior/precedent cases and statutes given a factual scenario. There were two tasks for this track: (i) Identifying relevant prior cases for a given situation (Precedent Retrieval), and (ii) Identifying most relevant statutes for a given situation (Statute Retrieval). Given a situation that can lead to filing a case, the precedent retrieval task aims at finding case documents where similar legal situations were addressed. The statute retrieval task aims at finding relevant statutes that are applicable to the situation. The factual scenarios, statutes and prior case documents used in the tasks were from the Indian Supreme Court judiciary.

Keywords: Legal data analytics · Prior case retrieval · Statute retrieval · Legal facts

1 Introduction

In countries following the Common Law system (e.g. UK, USA, Canada, Australia, India), there are two primary sources of law – Statutes (established laws, such as the Constitution of a country) and Precedents (prior cases decided in courts of law). Statutes deal with applying legal principles to a situation (facts / scenario / circumstances which lead to filing the case). Precedents or prior cases help a lawyer understand how the Court has dealt with similar scenarios in the past, and prepare the legal reasoning accordingly.

When a lawyer is presented with a situation (that will potentially lead to filing of a case), it will be very beneficial to him/her if there is an automatic system that identifies a set of related prior cases involving similar situations as well as statutes/acts that can be most suited to the purpose in the given situation.

Such a system shall not only help a lawyer but also benefit a common man, in a way of getting a preliminary understanding of the legal aspects pertaining to a situation, even before he/she approaches a lawyer. The system can assist him/her in identifying where his/her legal problem fits, what legal actions he/she can proceed with (through statutes) and what were the outcomes of similar cases (through precedents).

Motivated by the above scenario, we proposed two tasks in the FIRE 2019 track on 'Artificial Intelligence for Legal Assistance' (AILA): (1) Identifying relevant prior cases for a given situation (Precedent Retrieval), and (2) Identifying most relevant statutes for a given situation (Statute Retrieval). These tasks are described below.

1.1 Task 1: Identifying Relevant Prior Cases

The participants are given a set of 50 queries, each of which describes (in natural English language) a situation that had led to filing a case in an Indian court of law. We also provided 2,914 prior case documents that were judged in the Supreme Court of India. For each query, the task was to retrieve the most similar / relevant case documents with respect to the situation in the given query. Here, the concept of 'relevance' may be understood as follows – a prior case is considered relevant to a query if the case discusses a situation similar to that in the query, as judged by law experts.

1.2 Task 2: Identifying Relevant Statutes

We identified a set of 197 statutes (Sections of Acts) from Indian law, that are relevant to some of the queries stated above. We provided the participants with the title and description of these statutes. For each query, the task is to identify the most relevant statutes (from among the 197 statutes).

Both tasks consider Indian legal documents, i.e., Indian statutes and prior cases decided by Indian courts of Law (the dataset is detailed in the next section). Note that some similar research has been done on Chinese legal case judgments [12, 13], where state-of-the-art deep learning models have been applied to identify statutes given the facts of a case. It is to be noted that Chinese legal documents are very well-structured and segmented into section titles (similar to research papers). So the facts of the case can be easily extracted. On the other hand, Indian legal case judgments are not written in a structured way, and there are no section titles either. Hence it becomes a challenging task to extract the facts of the case [1] and create the datasets for these tasks.

2 Dataset

We collected all Indian Supreme Court case documents from 1952 to February 2018. We also collected 10,685 Acts (e.g., Constitution of India 1950, Indian

Penal Code 1860, Dowry Prohibition Act, 1961). Each Act contains articles or sections (e.g. Article 15 of the Constitution of India 1950, Section 302 of the Indian Penal Code 1860, Section 4 of the Dowry Prohibition Act, 1961 etc.). We call these articles/sections as *statutes*. All these documents were collected from Thomson Reuters Westlaw India.[6]

2.1 Creating Pool of Statutes

References to statutes is common in legal case documents. We extract the statutes (section or article number) cited from any of the case documents collected. Specifically, the cited statutes are identified using the fact that statute citations usually follow one of (or a combination of) the following template patterns:

- [section or article number] of the [Act]
  e.g., Section 47 of the Code of Criminal Procedure, 1973
- [section or article numbers xx and yy] of the [Act]
  e.g., Articles 15 and 21 of the Constitution
- [section or article numbers xx, yy and zz] of the [Act]
  e.g., Section 23, 27 and 39 of the Income-tax Act, 1961
- [section or article numbers xx to yy] of the [Act]
  e.g., Sections 56 to 60 of the Customs Act, 1962

In some case documents, Acts were cited without a year; such citations brought in an ambiguity if the said Act had multiple versions in different years, e.g., Finance Act 1980 , Finance Act 1983 , Finance Act 1984, etc. We discard all mentions of Acts that occur in multiple years and has been cited by a document without the particular year information.

We did not attempt to handle co-reference. For instance, a case document may mention at the beginning: “the benefit of probation under Section 4 of the Probation of Offenders Act, 1958 (herein referred to as the Act)”. In some later part of the document, it may cite “Section 6 of the said Act”. Our heuristic based approach could not capture Section 6 of the Probation of Offenders Act, 1958. However, the citation ‘Section 4 of the Probation of Offenders Act’ was captured.

To judge the performance of this heuristic method of extracting citations, we conducted a manual evaluation on a small set of 20 documents. Our method could achieve a precision of 1.0 and recall of 0.9 on this set. In other words, all the cited statutes identified by our method were correctly identified, and the method could correctly identify 90% of all cited statutes.

We then identified the top 200 most frequently cited statutes. Out of these, three (03) statutes were removed since they are repealed now. The resulting set contained 197 statutes. We anonymized the name of these statutes (e.g. Section 302 of the Indian Penal Code, 1860 was replaced with the identifier ‘S43’). Each statute has a title and description, e.g., the title of the Section 302 of the Indian

[6] http://www.westlawindia.com/. Note that we use only the publicly available full text judgement. All other proprietary information had been removed.

Penal Code, 1860 is *'Punishment for murder'*, and its corresponding description is *'Whoever commits murder shall be punished with death, or [imprisonment for life], and shall also be liable to fine'*. For each statute, the identifier, title and description were provided to the participants (for Task 2).

2.2 Query Formulation

We then identified the case documents which cite any of these 197 statutes (described above). From this set of case documents, we randomly selected 50 documents and extracted the facts manually. By 'facts' we mean, the chronology of events that led to filing the case. Next, we anonymize these facts to form the queries. Names, locations, etc. were replaced with generic abbreviations like 'P' (for the name of a person), 'L' (for the name of a place) etc. Dates and statute mentions (if any) were also removed. This was done to make the query look as generic as possible, while preserving the legal fact of the matter for proper statute and precedent retrieval. The queries thus obtained were used for both Task 1 and Task 2.

2.3 Creating Pool of Prior Case Documents

We next extract the prior cases cited in the 50 documents (from where the queries are formulated, as described above). If a case title along with its unique ID is mentioned in the current case document, then the case is considered as a prior case. The set of all prior cases for all the 50 documents were given as the pool of documents from where the relevant prior case with respect to a particular query had to be retrieved. The total number of prior cases (out of which relevant cases had to be retrieved in Task 1) is $2,914$.

2.4 Training and Test Data

Note that the tasks can be modeled either as unsupervised retrieval tasks (where one searches for relevant statues/prior cases) or as a supervised classification task (where one tries to predict for each statute/prior case whether it is relevant to the given query).

To facilitate considering the tasks as supervised learning tasks, we provided the gold standard sets for 10 queries to the participants as training data. The evaluations were done on the remaining 40 queries.

In summary, the queries (for both tasks) were derived from the facts stated in certain Supreme Court cases, and the gold standard results consisted of prior cases (for Task 1) and statutes (for Task 2) that were actually cited by the lawyers arguing those cases. Hence, the gold standard can be thought of as curated by law experts. We followed this automated methodology in creating the dataset, since actually involving legal experts (e.g., to find relevant prior cases / statutes) would have required a significant amount of financial resources and time.

3 Methodologies for Task 1: Precedent Retrieval

For the first task of retrieving relevant prior / precedent cases, we received a total of 23 runs from nine (09) participating teams. We briefly describe below the methodologies used by each team in each of their runs. The comparative results are in Table 1.

- **IITP**[2]: The team has its members from Government College Of Engineering and Textile Technology, Berhampore and Indian Institute of Technology Patna (both from India). They submitted two runs *IITP_BM25_case* and *IITP_doc2Vec_case*. Initially, every word from all the query and candidate case documents was converted to lowercase, lemmatized and stopwords were removed.
  In *IITP_BM25_case*, BM25 score between each query and each candidate were calculated. Top 2 documents retrieved based on this score were considered relevant.
  In *IITP_doc2Vec_case*, document vectors were learned using Doc2Vec. Then a normalized score between each query and each candidate were calculated and top 2 documents were retrieved based on this score.
- **HLJIT2019-AILA**[14]: This team was from Heilongjiang Institute of Technology, China. They submitted three runs that are described below. Their basic approach was to create an index of the case documents and try to extract keywords as query using different methods (runs). In the first run *HLJIT2019-AILA_task1_1*, top 50% of the IDF as the search key for each query was extracted. Then BM25 was used as the search score. In the second run *HLJIT2019-AILA_task1_2*, for each query, top 50% of the first IDF and the entire query as the search keywords was extracted. Then BM25 was used as the retrieval model. The search results were weighted and reordered. In the third method *HLJIT2019-AILA_task1_3*, the first 50% of the term and case documents of the TF-IDF for each query was extracted. Then word2vec was used to represent the vector. Euclidean distance was used to rank the documents.
- **HGC**[3]: This team had its members from Harbin Engineering University, Harbin, China and Heilongjiang Institute of Technology, Harbin, China. Their evaluation method was mainly based on document keywords. Firstly, they used TF-IDF and textRank to extract keywords from queries. Then they use space vector model, language model and BM25 model to retrieve keywords. Among them, the number of topic words was selected experimentally. They used the data of FIRE 2017 IRLeD Track [6] to conduct parametric experiments. By designing the keyword extraction method and the corresponding retrieval model for the experimental control group, they used the space vector model combined with TF-IDF method to retrieve the results, and selected the top 70 keywords in the query documents – this was their first run *HGC_1*. For their second run *HGC_2*, they used TF-IDF combined with BM25 model to retrieve, and the number of keywords is set

to 70. For their third run *HGC_3*, they used language model combined with textRank keyword extraction method to retrieve the first 60 words as query keywords.

– **Kavya S - Ganesh**[9]: This team was from Vellore Institute of Technology, Chennai, India. Their approach was based on computing TF-IDF vectors on the queries, statutes and case documents. The case documents were ranked based on the cosine similarity between the query and the case documents.

– **SSN_NLP**[4]: This team was from SSN College of Engineering, India. They used different query and document embeddings for the different runs. In the first run (*SSN_NLP_1*), they used Word2Vec embeddings. In the second run (*SSN_NLP_2*), they concatenated Word2vec and Glove vectors. In the third run (*SSN_NLP_3*), they used the TF-IDF measure.

– **TRDDC Pune**[8]: This team had its members for College of Engineering Pune and TRDDC Pune, India. From each query, they extracted sentences containing the 'appeal' and 'appeal'-related part. From the extracted part, they removed named entities, stop-words and punctuation. Next they pre-processed the data. First they split each document into paragraphs. From each paragraph, they removed named entities, stop-words and punctuation marks. For retrieval, they use TF-IDF, BM-25 and an Ensemble (of TF-IDF and BM-25). For each run they use all the case documents as corpus. They calculated the score of fact query vs every paragraph in case document. The final score of each document is calculated by taking average of score of top 3 paragraphs of that document.

– **CUSAT-NLP**[10]: This team was from Cochin University of Science and Technology (CUSAT), India. In their first run (*Task1_CUSAT_NLP_1*), the corpus of case documents were trained using Doc2Vec. Each case and query document are represented as a 100-dimensional vector. Cosine similarities between each query and case documents are calculated and sorted to obtain the most similar prior cases. In the second run (*Task1_CUSAT_NLP_2*), the corpus used for training Doc2Vec is the complete list of case documents, query documents and statutes documents. Rest of the steps were the same as run1.

– **JU-SRM**[7]: This team was from SRM University, Chennai and Jadavpur University, Kolkata (both from India). They used sentence level (Sent2Vec) and word level (Word2Vec) representations of the documents and queries. In the first run, they used pre-trained Sent2Vec model to represent the case documents and queries. In the second run, a FastText model was trained on 2914 case documents, and this model was used for representation. For both runs, for each query-case pair, cosine-similarity was calculated. Using the cosine-similarity score, the rankings were calculated.

– **thuir_legal**[11]: This team was from Tsinghua University, Beijing, China. In the first run (*thuir_legal_1*), they used the language model based retrieval. They have used unigram and bigram with linear interpolation. In the second run (*thuir_legal_2*), they use vector space model (TF-IDF) for retrieval. In the third run (*thuir_legal_3*), they combined the vector space model and mixed size bigram model using min-max normalization.

4 Methodologies for Task 2: Statute Retrieval

For the second task of retrieving relevant statutes , we received a total of nineteen (19) runs from eight (08) participating teams. We briefly describe below the methodologies used by each team in each of their runs. The comparative results are in Table 2.

- **IITP**[2]: The team has its members from Government College Of Engineering And Textile Technology, Berhampore and Indian Institute of Technology Patna. In their run, all query and statutes were taken and from every document, every word converted into lowercase, stopwords were removed and words were lemmatized. Then BM25 score between each query and each statute was calculated, and the statutes sorted as per decreasing order of the score. Then the score was normalized and top 3 statutes were considered relevant for each query.
- **HLJIT2019-AILA**[14]: This team was from Heilongjiang Institute of Technology, China. They created an index on the statutes and try to reorder the results using different methods. In the first run, they extracted the top 50% of the IDF as the search key for each query of Query_doc, and used BM25 as the search score. In the second method, for each query, the top 10 of the search result of the first method in the first task is used as the candidate query, and each case is extracted as the search keyword of the first 50% of the IDF. Searched in the statutes index, the document frequency of each case retrieval result is used as the sorting method. In the third method, weighting and reordering the search results of each case in the second method was performed.
- **SSN_NLP**[4]: This team was from SSN College of Engineering, and submitted three runs. They considered a case and a statute document and then used (i) TF-IDF, (ii) Jaccard Similarity and (iii) count of common words in their three different runs respectively.
- **Kavya S - Ganesh**[9]: This team was from Vellore Institute of Technology, Chennai. Their approach was based on computing TF-IDF vectors on the queries, statutes and case documents. The statutes were ranked based on the cosine similarity between the query and the statute documents.
- **CUSAT-NLP**[10]: This team was from Cochin University of Science and Technology (CUSAT). In run 1, the corpus of case documents were trained using Doc2Vec. Each case and query document were represented as a 100-dimensional vector. Cosine similarities between each query and case documents are calculated and sorted to obtain the most similar prior cases. In the second run,the corpus used for training Doc2Vec is the complete list of

case documents, query documents and statutes documents. Rest of the steps were the same as the first run.

- **JU-SRM**[7]: From queries, key-phrases were extracted using *rake-nltk* library. From statutes, key-phrases were extracted using the same library. Then manual editing was done (addition and removal). Each query and statute key-phrases were encoded using pre-trained BERT model (with bert-embedding package).
  In the first run, for each query statute pair, cosine-similarity scores of each query_key_phrase and statute_key_phrase was calculated. Then highest and second highest cosine similarity scores for each query-statute keyphrase pair were multiplied. This value was used as the similarity score between query statute pair. Using these values, rankings were calculated.
  In the second run, for each query statute pair, cosine-similarity scores of each query_key_phrase and statute_key_phrase was calculated. Then highest query-statute keyphrase cosine-similarity was calculated (suppose H). Sum(all cosine similarity scores of query-statute keyphrase)/(no. of keyphrase in query $\times$ no. of keyphrase in statute) was calculated (suppose AVG). Based on this similarity score the rankings were decided.
  In the third run, along with highest (H) the second highest (SH) was also considered. The similarity score was then calculated as $H \times SH \times AVG$, which was used for ranking.

- **UBLTM**[5]: This team was from the University of Botswana. They investigated the effect of different retrieval models using Terrier 4.2. For the first run they transform both the documents and statutes into TREC style format and use IFB2 weighting model of Terrier. For the second run they used only the title field of statute on IFB2. In the third run, they used only the description field of statute to retrieve using IFB2. The idea was to test the retrieval effectiveness of the both the fields and each of the two fields.

- **thuir_legal**[11]: This team was from Tsinghua University, Beijing, China. They first generate summary of the query automatically instead of extracting the key and context sentences. The summarization tool is PKUSUMSUM, and they used the LexPageRank algorithm. They apply the restriction that the generated summary of one query contains no more than 200 words. As for statutes, they use both the title and description contents. In the first run, they used the language model based retrieval. They have used unigram and bigram with linear interpolation. In the second run, they used vector space model (TF-IDF) for retrieval. In the third run, they combined the vector space model and BM25 using weighted averaging their scores after min-max normalization. The final weights were the ones that performed best on the training data.

5 Evaluation

In both tasks, a submitted method generated a ranked list of documents (prior cases for Task 1 and statutes for Task 2) relevant to each query. One set of ranked lists (one for each query) generated by a certain method is called a 'run'. We evaluate the submitted runs, based on their performance over all 40 test queries. To this end, we used the following performance measures:

– **Mean Average Precision (MAP)**: mean of the average precision scores for each query. This is our primary measure based on which the different runs are ranked.
– **Precision at 10 (P10)**: Number of relevant documents in the top 10 ranked results, averaged over all queries. Since each query contains 10 precedents and statutes on average, we report this score for the runs.
– **BPREF** [7]: a summation-based measure of how many relevant documents are ranked before irrelevant documents, averaged over all queries. This measure is especially useful when the set of relevant documents may not be completely known (as can be the case here).
– **recip_rank** [8]: inverse of the position of the *first* relevant document in the ranked list of a query, averaged over all queries.

We used the *trec_eval* tool [9] for computing the metrics stated above. We choose MAP as the primary measure since it incorporates both Precision and Recall.

6 Results

Task 1: Results for Task 1 (retrieving prior case documents given a factual situation as query) is presented in Table 1. The runs are sorted in decreasing order of the MAP scores. The best performing run achieves MAP of 0.1492. This relatively low MAP score reflects that the task is challenging.

Note that the methods are all unsupervised. The training data was used mainly to understand how an approach was performing and for tuning parameters (e.g., assigning weights to different models when an ensemble of models were considered). Embedding methods like FastText, Sent2Vec were also used but the performance was not good. This is probably because these methods require a huge amount of data to be trained on. Additionally, the Legal domain being very specialized, pre-trained open-domain models do not perform well.

Task 2: Results for the task of retrieving relevant statutes given the facts is presented in Table 2. The runs are sorted in decreasing order of MAP scores. The best performing method achieves 0.1566, which again reflects the challenging

[7] https://trec.nist.gov/pubs/trec16/appendices/measures.pdf
[8] https://www-nlpir.nist.gov/projects/trecvid/trecvid.tools/trec_eval_video/A.README
[9] https://trec.nist.gov/trec_eval/

Table 1. Results of Task 1: Precedent retrieval for queries. All measures averaged over 40 test queries. Rows are sorted in decreasing order of MAP score (primary measure).

Team name	Run ID	P@10	MAP	BPREF	recip_rank	Method Summary
HLJIT2019-AILA	HLJIT2019-AILA_task1_2	0.07	0.1492	0.1286	0.288	IDF, BM 25, weighted re-ordering of results
HGC	HGC_1	0.0575	0.1382	0.1207	0.28	TF-IDF
HLJIT2019-AILA	HLJIT2019-AILA_task1_1	0.06	0.1335	0.1134	0.282	IDF, BM25
HGC	HGC_2	0.05	0.1263	0.1092	0.256	TF-IDF, BM25
IITP	IITP_BM25_case	0.0275	0.0984	0.0869	0.175	BM25, top 2 docs considered relevant
TRDDC Pune	TFIDF	0.05	0.0956	0.067	0.203	TFIDF
HGC	HGC_3	0.0316	0.0946	0.0804	0.18	Language model, textRank
TRDDC Pune	ENSEMBLE	0.04	0.0817	0.0591	0.162	ensemble of TF-ID, FBM25
TRDDC Pune	BM25	0.0375	0.0773	0.0547	0.151	BM25
thuir_legal	thuir_legal_3	0.0425	0.0689	0.0434	0.121	VSM, mixed size bigram model
IITP	IITP_Doc2Vec_case	0.0175	0.0677	0.0552	0.138	doc2vec embeddings
thuir_legal	thuir_legal_1	0.0375	0.0599	0.0316	0.149	language model
CUSAT_NLP	Task1_CUSAT_NLP_1	0.03	0.0481	0.0412	0.166	doc2vec embeddings
JU_SRM	JU_SRM_1	0.025	0.0478	0.0284	0.131	pre trained Sent2Vec model
Kavya S Ganesh	R1	0.01	0.0416	0.0131	0.069	TF-IDF
SSN_NLP	SSN_NLP_1	0.03	0.0405	0.0277	0.091	Word2Vec embeddings
thuir_legal	thuir_legal_2	0.0225	0.0405	0.0221	0.095	VSM
CUSAT_NLP	Task1_CUSAT_NLP_2	0.02	0.0264	0.0227	0.102	doc2vec embeddings
JU_SRM	JU_SRM_2	0.0175	0.0228	0.0163	0.065	FastText
HLJIT2019-AILA	HLJIT2019-AILA_task1_3	0.015	0.022	0.0066	0.065	TF-IDF, Word2Vec
JU_SRM	JU_SRM_3	0.02	0.0181	0.006	0.044	Sent2Vec, FastText
SSN_NLP	SSN_NLP_2	0	0.0026	0	0.003	Word2Vec, Glove
SSN_NLP	SSN_NLP_3	0	0.0025	0	0.003	TF-IDF

nature of the task. Similar to Task 1, the methods were mostly unsupervised except one where keyphrases were extracted manually (JU_SRM_6).

Note that, the statute descriptions are usually smaller in length than the factual queries supplied in this task. Hence the idea of summarizing the query (as done by the best performing run) seems to be promising.

Although MAP is considered as the primary metric here, it can be noted that *recip_rank* also plays an important role. The tasks being very challenging, it is useful to understand at what position in the ranked list the first relevant document comes up. For both Tasks 1 and 2, we see that this score is 0.28 approximately.

7 Concluding Discussions

The FIRE 2019 AILA track has successfully created a benchmark collection of factual queries and its relevant statutes and prior cases. As evident from the result tables, we find that the tasks of understanding what statutes and precedent cases can be relevant to a given situation, is indeed a challenging task. The best performing methods achieve a MAP score of 0.1492 and 0.1566 respectively on the tasks, which shows that there is lot of scope of improvement.

Table 2. Results of Task 2: Statute retrieval for queries. Measures averaged over 40 test queries. Rows are sorted in decreasing order of MAP score (primary measure).

Team name	**Run ID**	**P @ 10**	**MAP**	**BPREF**	**recip_rank**	**Method Summary**
thuir_legal	thuir_legal_2	0.0975	0.1566	0.0961	0.281	query summarization, language model
thuir_legal	thuir_legal_3	0.09	0.1318	0.0742	0.247	VSM
thuir_legal	thuir_legal_1	0.065	0.1115	0.0653	0.23	VSM + BM25
UBLTM	UBLTM2	0.0725	0.1023	0.0571	0.211	IFB2 on statute titles only
UBLTM	UBLTM3	0.0725	0.1023	0.0571	0.211	IFB2 of statute description only
UBLTM	UBLTM1	0.0725	0.1022	0.0571	0.214	IFB2 of Terrier on whole statute
CUSAT_NLP	Task2_CUSAT_NLP_2	0.055	0.0967	0.0377	0.199	Doc2vec
JU_SRM	JU_SRM_5	0.06	0.0918	0.0402	0.201	manual keyphrase extraction ; BERT
CUSAT_NLP	Task2_CUSAT_NLP_1	0.055	0.0866	0.0412	0.202	Doc2vec
JU_SRM	JU_SRM_6	0.06	0.0831	0.0285	0.162	manual keyphrase extraction ; BERT
HLJIT2019-AILA	HLJIT2019-AILA_task2_3	0.0675	0.0819	0.0703	0.279	TF-IDF, BM25
SSN_NLP	SSN_NLP_2	0.0475	0.0778	0.0494	0.191	Jaccard Similarity
HLJIT2019-AILA	HLJIT2019-AILA_task2_2	0.0675	0.0773	0.0671	0.263	TF-IDF
JU_SRM	JU_SRM_4	0.06	0.0767	0.0309	0.146	manual keyphrase extraction ; BERT
Kavya S Ganesh	R1	0.035	0.0682	0.054	0.136	TF-IDF
Kayalvizhi S - SSN_NLP	SSN_NLP_3	0.03	0.0611	0.0317	0.125	Count vectors
HLJIT2019-AILA	HLJIT2019-AILA_task2_1	0.0675	0.0606	0.0516	0.2	TF-IDF, BM25
SSN_NLP	SSN_NLP_1	0.025	0.0518	0.0285	0.128	TF-IDF
BIITP	IITP_BM25_statutes	0.02	0.036	0.0397	0.129	BM25

In future, we plan to extend the gold standard citations of statutes and precedents. It is possible that there are certain statutes and prior case documents that are actually *relevant* to the situation but *has not been cited / made use of* in the judgements. However, for a legal practitioner/layman, knowing the whole set of relevant documents is important, and not only the ones specifically mentioned in the Supreme Court case judgment. Hence we plan to extend the gold standard, e.g., through pooling from the top results returned by the runs. However, this extension will require involvement of domain experts for judging relevance of the pooled documents, which will require significant amount of time (due to which we could not do the pooling in the time frame of this FIRE track).

Acknowledgements: The track organizers thank all the participants for their interest in this track. We also thank the FIRE 2019 organizers for their support in organizing the track.

References

1. Bhattacharya, P., Hiware, K., Rajgaria, S., Pochhi, N., Ghosh, K., Ghosh, S.: A Comparative Study of Summarization Algorithms Applied to Legal Case Judgments. In: Advances in Information Retrieval – Proceedings of European Conference on Information Retrieval (ECIR). pp. 413–428 (2019)
2. Gain, B., Bandyopadhyay, D., De, A., Saikh, T., Ekbal, A.: IITP at AILA 2019: System Report for Artificial Intelligence for Legal Assistance Shared Task. In:

Proceedings of FIRE 2019 - Forum for Information Retrieval Evaluation (December 2019)

3. Gao, J., Ning, H., Sun, H., Liu, R., Han, Z., Kong, L., Qi, H.: Fire2019@aila: Legal retrieval based on information retrieval model. In: Proceedings of FIRE 2019 - Forum for Information Retrieval Evaluation (December 2019)
4. Kayalvizhi, S., Thenmozhi, D., Aravindan, C.: Legal assistance using word embeddings. In: Proceedings of FIRE 2019 - Forum for Information Retrieval Evaluation (December 2019)
5. Lefoane, M., Koboyatshwene, T., Rammidi, G., Narasimham, V.L.: Legal statutes retrieval: A comparative approach on performance of title and statutes descriptive text. In: Proceedings of FIRE 2019 - Forum for Information Retrieval Evaluation (December 2019)
6. Mandal, A., Ghosh, K., Bhattacharya, A., Pal, A., Ghosh, S.: Overview of the fire 2017 irled track: Information retrieval from legal documents. In: Working notes of Annual Meeting of the Forum for Information Retrieval Evaluation (FIRE) – CEUR workshop proceedings, Volume 2036. pp. 63–68 (2017)
7. Mandal, S., Das, S.D.: Unsupervised identification of relevant cases & statutes using word embeddings. In: Proceedings of FIRE 2019 - Forum for Information Retrieval Evaluation (December 2019)
8. More, R., Patil, J., Palaskar, A., Pawde, A.: Removing Named Entities to Find Precedent Legal Cases. In: Proceedings of FIRE 2019 - Forum for Information Retrieval Evaluation (December 2019)
9. Rameshkannan, R., Rajalakshmi, R.: Dlrg@aila 2019:context - aware legal assistance system. In: Proceedings of FIRE 2019 - Forum for Information Retrieval Evaluation (December 2019)
10. Renjit, S., Idicula, S.M.: Cusat nlp@aila-fire2019: Similarity in legal texts using document level embeddings. In: Proceedings of FIRE 2019 - Forum for Information Retrieval Evaluation (December 2019)
11. Shao, Y., Ye, Z.: THUIR@AILA 2019: Information Retrieval Approaches for Identifying Relevant Precedents and Statutes. In: Proceedings of FIRE 2019 - Forum for Information Retrieval Evaluation (December 2019)
12. Wang, P., Fan, Y., Niu, S., Yang, Z., Zhang, Y., Guo, J.: Hierarchical matching network for crime classification. In: Proceedings of ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 325–334. SIGIR'19 (2019)
13. Wang, P., Yang, Z., Niu, S., Zhang, Y., Zhang, L., Niu, S.: Modeling dynamic pairwise attention for crime classification over legal articles. In: Proceedings of ACM SIGIR Conference on Research & Development in Information Retrieval. pp. 485–494. SIGIR '18 (2018)
14. Zhao, Z., Ning, H., Huang, C., Kong, L., Han, Y., Han, Z.: Fire2019@aila: Legal information retrieval using improved bm25. In: Proceedings of FIRE 2019 - Forum for Information Retrieval Evaluation (December 2019)