Overview of the FIRE 2017 IRLeD Track: Information Retrieval from Legal Documents

Arpan Mandal
IIEST Shibpur
India

Kripabandhu Ghosh
IIT Kanpur
India

Arnab Bhattacharya
IIT Kanpur
India

Arindam Pal
TCS Research
India

Saptarshi Ghosh
IIT Kharagpur; IIEST Shibpur
India

ABSTRACT

The FIRE 2017 IRLeD Track focused on creating a framework for evaluating different methods of Information Retrieval from legal documents. There were two tasks for this track: (i) Catchphrase Extraction task, and (ii) Precedence Retrieval task. In the catchphrase extraction task, the participants had to extract catchphrases (legal keywords) from Indian Supreme Court case documents. In the second task of Precedence Retrieval, the participants were to retrieve relevant or cite-able documents for particular Indian Supreme Court cases from a set of prior case documents.

CCS CONCEPTS

•Information systems →Information retrieval;

KEYWORDS

Legal Information Retrieval, Prior Case Retrieval, Legal Catchphrase Extraction

1 INTRODUCTION

In a *Common Law System*[1], great importance is given to prior cases. A prior case (also called a precedent) is an older court case related to the current case, which discusses similar issue(s) and which can be used as reference in the current case. A prior case is treated as important as any law written in the law book (called statutes). This is to ensure that a similar situation is treated similarly in every case. If an ongoing case has any related/relevant legal issue(s) that has already been decided, then the court is expected to follow the interpretations made in the prior case. For this purpose, it is critical for legal practitioners to find and study previous court cases, so as to examine how the ongoing issues were interpreted in the older cases.

With the recent developments in information technology, the number of digitally available legal documents has rapidly increased. It is, hence, imperative for legal practitioners to have an automatic precedent retrieval system. The task of precedence retrieval can be modeled as a task of information retrieval, where the current case document (or a description of the current situation) will be used as the query, and the system should return relevant prior cases as results.

Additionally, legal texts (e.g., court case descriptions) are generally long and have complex structures [4]. This nature makes their thorough reading time-consuming and strenuous, even after relevant cases are retrieved. So, apart from a precedence retrieval system, it is also essential for legal practitioners to have a concise representation of the core legal issues described in a legal text [10]. One way to list the core legal issues is by keywords or key phrases, which are known as 'catchphrases' in the legal domain [6].

Motivated by the requirements described above, The IRLeD track focused on the following two tasks: Catchphrase extraction, Precedence retrieval.

1.1 Task 1: Catchphrase Extraction

Catchphrases are short phrases from within the text of the document. Catchphrases can be extracted by selecting certain portions from the text of the document.

In this task, a set of 400 legal documents (Indian Supreme Court case documents) was provided to the participants. For 100 of these documents (training set), gold standard catchphrases were provided – these gold standard catchphrases were obtained from a well-known legal search system *Manupatra* (https://www.manupatra.com/), which employs legal experts to manually annotate case documents with catchphrases. The rest 300 documents were used as the test set. The participants were expected to extract the catchphrases for the documents in the test set.

1.2 Task 2: Precedence Retrieval

For this task, two sets of documents were provided:
(1) Current cases: A set of 200 Indian Supreme Court cases, for which the prior cases were to be retrieved.
(2) Prior cases: For each *current case*, we obtained a set of prior cases that were actually cited in the case decision. 1000 such cited prior cases were present in the second set of documents, along with other 1000 documents which were *not* cited from any document in the 'current cases' set.

For each document d in the first set (current cases), the participants were to return a ranked list of documents from the second set (prior cases), in a way that the cases that were actually cited from d are ranked higher than the other documents (that were not cited from d).

2 DATASET

We have developed two datasets corresponding to the two tasks:

(1) Data for Task 1: A collection of legal case documents with their catchphrases: We built a dataset containing 400 court case

[1] https://en.wikipedia.org/wiki/Common_law/ as seen on 6th November, 2017.

Case Id	Catchphrases
1953.INSC.24 http://liiofindia.org/in/cases/cen/INSC/1953/24.html	Actual Delivery, Advocate-General, Alternative Remedy, Appropriate, Assessment, Carrier, Carrying on Business, Cause of Action, Commencement of the Constitution, Competent Legislature, Consignment, Constitution of India, Constitutional Validity, Consumption, Contract, Contract of Sale, Contravention, Cost, Dealer, Declared by Parliament, Deduction, Definition, Delegate, Demand Notice, Despatch, Discrimination, Discriminatory, Double Taxation, Existing Law, Export, Federal Court, Freedom of Trade
1991.INSC.12 http://liiofindia.org/in/cases/cen/INSC/1991/12.html	Advisory Board, Allowance, Appropriate Government, Arrest, Constitutional Obligation, Constitutional Question, Constitutional Safeguard, Detaining Authority, Detention, Detenu, Duty of the State, Earliest Opportunity, General Clauses Act, Grounds of Detention, Guarantee, Legal Obligation, Liberty, Order of Detention
1983.INSC.27 http://liiofindia.org/in/cases/cen/INSC/1983/37.html	Commutation, Confinement, Conspiracy, Constitution of India, Death Sentence, Fundamental Right, Imposition of Death Sentence, Judicial Proceeding, Life Imprisonment, Solitary Confinement, Speedy Trial, Transportation for Life

Table 1: Examples of Indian Supreme Court cases and catchphrases taken from the Manupatra legal expert system (reproduced from [6])

documents of the Indian Supreme Court, along with their catchphrases. The texts and their catchphrases were obtained from a well-known legal search system *Manupatra* which uses human legal experts to annotate court case documents with catchphrases. All decisions and the corresponding Catchphrases are available in text format. A few example Catchphrases are shown in Table 1 (reproduced from [6]).

The collection provided for the track consisted of 400 Indian Supreme Court case documents. Out of these, 100 documents were provided along with their gold standard catchphrases (training set) while the participants were expected to find the catchphrases for the rest 300 documents (test set).

(1) Data for Task 2: A collection of legal case documents, and prior cases cited from them: We crawled a large number of case documents of cases judged at the Supreme Court of India, from the site *LIIofIndia* (www.liiofindia.org/).[2] The documents were downloaded in HTML, and the downloaded HTML files were then parsed to get the final texts.

The dataset for the task contained 1000 current (query) cases that were judged after the year 2000, and 2000 prior cases that were judged prior to the year 2000 (as described in the Introduction). All filenames were anonymized, and all citation markers from the current/prior cases were replaced with a special marker.

3 METHODOLOGIES FOR TASK 1: CATCHPHRASE EXTRACTION

For the first task of Catchphrase extraction, we received a total of ten runs from seven participating teams. All the runs were supervised in nature except the run *UBIRLeD_1*, as described in Table 2. We briefly describe below the methodologies used by each team in each of their runs.

- **rightstepspune:** This team participated from Right Steps Consultancy, Pune. In the method in their only run, the problem of catchphrase detection was modeled as sequential probabilistic labeling problem rather than a simple linear classification problem. Conditional Random Fields (CRF) algorithm was chosen with primary features such as POS (part-of-speech) and custom NER (Named Entity Recognition) tags and numerous secondary features representing the context. They first tokensied the texts into tokens using NLTK[3] tokenizer. Then they applied POS (part-of-speech) tags to each of the tokens again using the NLTK toolkit. These features along with several other features were used to train a model of CRF, which was then used to predict the catchphrases.
- **UBIRLED:** This team participated from the University of Botswana, Computer Science Department. They submitted two runs. For this they have used two recently developed catchphrase extraction tools:
  1. **RAKE (Rapid Automatic Keyword Extraction):** an unsupervised algorithm for keyword extraction [13].
  2. **MAUI:** a supervised algorithm for keyword extraction [8].
- **AMRITA_CEN_NLP:** This team participated from Amrita Vishwa Vidhyappetham, India and submitted a supervised and fully automatic run. For this they have first determined a set of candidate catchphrases and hence represented the documents and candidate catchphrases as vectors. They used *Doc2Vec*[5] for representing the texts as vectors. Hence the scoring of candidate catchphrases was simply done by measuring the cosine similarity of their vector with the document vector.
- **HLJIT2017:** This team participated from the Heilongjiang Institute of Technology, China. They have submitted three runs in total. In all the three methods, they have approached the task as a classification problem and have used supervised fully automatic techniques.

  For the first two runs they used bagging techniques. Here, the training set is divided into different sampling sets. Then these sampling sets are hence used to train a base classifier. They considered the base classifier as *Decision tree*[11] in one run and *Random forest*[3] in another run. In the third run they have used *RankSVM*[4] which

[2]LIIofIndia is a website hosted by the Legal Information Institute of India.

[3]http://www.nltk.org/

[4]urlhttp://www.cs.cornell.edu/People/tj/svm_light/svm_rank.html

Run_ID	R-Prec	Prec@10	Recall@100	MAP	Overall Recall	Method Summary
rightstepspune_1_task1	0.215	0.281	0.248	0.479	0.248	CRF, POS, NER
UBIRLeD_2	0.190	0.254	0.305	0.370	0.326	MAUI[8]
AMRITA_CEN_NLP_RBG1_1	0.168	0.144	0.535	0.200	0.652	Doc2Vec
HLJIT2017_IRLeD_Task1_3	0.086	0.122	0.151	0.165	0.152	RankSVM
bphc_withPOS_1	0.066	0.102	0.137	0.161	0.165	TF, POS tags
HLJIT2017_IRLeD_Task1_1	0.030	0.058	0.033	0.140	0.033	Decision Tree
HLJIT2017_IRLeD_Task1_2	0.034	0.060	0.044	0.124	0.044	Random Forest
Bits_Pilani_1	0.030	0.049	0.080	0.093	0.100	LSTM network
FIRE_2017_SR	0.026	0.025	0.087	0.062	0.161	POS, Deep Neural Network
UBIRLeD_1	0.023	0.014	0.172	0.046	0.499	RAKE[13]

Table 2: Evaluation of runs for Task 1: Catchphrase Extraction. Runs are sorted in descending order of the Mean Average Precision.

uses Support Vector Machines (*SVM*) to solve the ranking problem of ranking the catchphrases.

- **bphc_withPOS_1:** This team participated from Birla Institute of Technology & Science, Pilani, India. They mainly concentrated on the preprocessing part and term scoring methods rather than phrase scoring methods. Their method extracts words rather than phrases. After a series of basic pre-processing, for scoring different unigrams they considered the frequency of occurrence within the document. Also, they have given a POS based weightage by checking which POS tags were more likely to be present within a catchphrase.
- **BITS_Pilani:** This team has participated from Birla Institute of Technology and Science, Pilani, India. They have submitted a supervised and fully automatic approach for extracting catchphrases.

  The problem is formulated as a classification task and the objective is to learn a classifier using LSTM network. The proposed methodology involves a pipelined approach and is divided into four phases:
  - Pre-processing
  - Candidate phrase generation
  - Creating vector representations for the phrases
  - Training a LSTM network
- **FIRE_2017_SR:** This team has participated from Indian Institute of Engineering Science and Technology, Shibpur, India. They have submitted one fully automatic supervised run. They used a deep neural network to train on a number of different features of the actual phrases. For extraction of catchphrases, a set of candidate phrases are first selected using POS (part-of-speech) tags of the known catchwords. Once the candidate phrases are obtained. These candidate phrases are then classified using the deep neural network already trained.

4 METHODOLOGIES FOR TASK 2: PRIOR CASE RETRIEVAL

In the second task of Precedence Retrieval, we received twenty one runs in total from nine participating teams. All of these runs were fully automatic in nature and their performance is as shown in Table 3. Described below are the methodologies used by each team in each of their runs.

- **flt_ielab:** This team participated from Queensland University of Technology, Australia. They submitted a total of three runs each of which use fully automatic methods. For each of the query documents they have formed a set of queries from the positions where the actual citations were present[5]. Now the query formation was differently done in the three runs as described below:
  (1) *flt_ielab_para:* Here, the query was formed by considering a paragraph around the citation marker.
  (2) *flt_ielab_idf:* Here, only 50% of the words were considered after weighing the terms by their *idf* (inverse document frequency).
  (3) *flt_ielab_plm:* Here, only 50% of the words given by *flt_ielab_idf* were considered by its probability from a parsimonious language model.

  Before applying the above filters to get the query terms, all terms were cleaned by removing stopwords and punctuation marks. Once the query terms were ready, they were used to retrieve prior cases using BM25[12] algorithm.

  As, a single query document has multiple citation markers. So, the final set of retrieved documents was chosen to be the top-scored 1000 documents from the union of retrieved documents by all these queries.
- **HLJIT2017_IRLeD_Task2:** This team from Heilongjiang Institute of Technology, China submitted three runs. Al of the runs were fully automatic in nature. The runs are described as follows:
  - **run_1:** In this run they have used a language model based on Dirichlet Prior Smoothing[14].
  - **run_2:** For the second search model they chose BM25 algorithm[12], which is a well-known probability based model.
  - **run_3:** In the third run they used *lucene*[7] which implements a vector space model to estimate the relevance of query and document.

[5]Note that the positions of the actual citations were marked using a marker in all the text documents.

- **SSN_NLP:** This team participated from SSN College of Engineering, India. They submitted three fully automatic runs as described below:
  - **run_1:** They considered the TF-IDF vectors of each document by using the TF-IDF vectorizer tool implemented in *scikit-learn*[6]. While considering the TF-IDF vectors they have considered only the nouns in the document. Now, cosine similarity between the query document and the set of prior cases are calculated and hence sorted to present the top scored documents.
  - **run_2:** This is very similar to the first run except that while calculating the TF-IDF vectors, verbs were also considered in addition to nouns.
  - **run_3:** This run considers Word2Vec vectors for each document in addition to the TF-IDF vectors as described in the second run.
- **rightstepspune_1_task2:** This team participated from RightSteps Consultancy, India. They submitted one fully automatic run. For measuring the similarity score between a pair of cases, they have used a weighed average of three different methods:
  - **Regular Expression based:** Here, different legal statutes (such as Articles) referred within the text were captured by using pattern matching. Once the list of statutes have been obtained for a given query document, the same is attempted for every prior cases. All prior cases that has any statutes in common are retrieved.
  - **Topic Modeling based:** In this method they employ the implementation of Latent Dirichlet Allocation (LDA) as in the *gensim* package.[7] Hence, score of similarity is calculated based on ratio of matching topic-words to the total.
  - **Using Document Vector:** To generate the document vectors the following steps were followed:
    1. Got every case as cleaned text, split it to form list of words/tokens, for both, current and prior cases.
    2. Created *gensim TaggedDocument* for each case text, giving filename as tag.
    3. A Map of tag to the content i.e. word-list for each cases were generated and saved for reuse.
    4. LDA model was built and saved. It was used to generate document vectors for both current and prior cases.

    A similarity matrix was generated where current cases are rows and prior cases as columns with values as cosine similarity between document vectors of the current-prior case pair (row-column). The values act as score for this particular approach.
- **UB_Botswana_Legal_Task2:** This team participated from University of Botswana, Botswana. They submitted three fully automatic runs. One common part in all the runs was the basic query formulation. For this, they have tokenized the text and removed all stopwords and stemmed them using Porter Stemmer. The nest steps for each run is described below:
  - **run_1:** Using the formulated queries, they have deployed the parameter-free DPH term weighting model from the Divergence from Randomness (DFR) framework[2] IR platform as our baseline system to score and rank the prior cases.
  - **run_2:** They used the first run as the baseline system. In addition, they deployed the Sequential Dependence (SD) variant of the Markov Random Fields for term dependence. Sequential Dependence only assumes a dependence between neighbouring query terms [9, 15]. In this work, they used a default window size of 2 as provided in Terrier-4.2.[8]
  - **run_3:** They used the first run as the baseline system. In addition, they deployed a simple pseudo-relevance feedback on the local collection. They used the *Bo1 model* [1] for query expansion to select the 10 most informative terms from the top 3 ranked documents after the first pass retrieval (on the local collection). They performed a second pass retrieval on this local collection with the new expanded query.
- **UBIRLeD:** This is another team participating from University of Botswana, Botswana. They have submitted three runs all of them being fully automatic in nature. For each of the runs they have retrieved 1000 ranked prior case judgments.

  For the second and third runs they have parsed the prior case documents into two parts. To identify the most informative terms they have used topic modeling, specifically Latent Dirichlet Allocation (LDA). The terms identified using LDA were then used to parse prior cases into documents with two fields:
  1. LDA_TEXT - A field containing words that have been identified as most informative words for the collection of prior case judgments.
  2. OTHER_TEXT - A field containing other words that have not been identified as most informative words.

  The runs are as described below:
  - **run_1:** A Baseline run where they have used the original dataset, only parsing it to TREC format, the runs were obtained using BM25 with default settings.
  - **run_2:** This is the run for a field based retrieval approach where the weight of LDA_TEXT was set to be far lower than the weight of OTHER_TEXT, specifically they have used BM25F weighting model, parameter settings for the weight assigned to LDA_TEXT and OTHER_TEXT is 0.2 : 1.0 in Terrier respectively, all other parameters were left as default.
  - **run_3:** This is the run for a field based retrieval approach where the weight of LDA_TEXT was set to be

[6]A open source library in python available at: http://scikit-learn.org/stable/
[7]*gensim* is a python package available at https://radimrehurek.com/gensim/.
[8]Terrier is an open source Information Retrieval platform available at http://terrier.org/.

far bigger than the weight of OTHER_TEXT, specifically they have used BM25F weighting model, parameter settings for the weight assigned to LDA_TEXT and OTHER_TEXT is 2.0 : 1.0 in terrier respectively, all other parameters were left as default.

- **bphcTask2IRLeD:** This team has participated from Birla Institute of Technology & Sciences, Pilani, India. They have submitted one run that is fully Automatic in nature. Here, they have considered a minimized set of words by considering only 5000 such words whose combined score of POS (part-of-speech) occurrence probability and IDF (inverse document frequency) score is higher than the rest of the words. Using this focused subset of words they have formed document vectors for each of the documents (both prior cases and current cases). Each vector is of size 5000, where each field corresponds to each word in the focused set. Now a vector for a document is so formed that if a word in the focused set is present then its value in the corresponding field is the combination of its TF (term frequency), IDF, and POS occurrence probability. Now, the similarity score between two document vectors are measured by simply finding the dot product of the two. For each Query Case, the similarity is calculated for between this and all prior cases. Then the top ranked prior cases are reported.
- **AMRITA_CEN_NLP_RBG:** This team has participated from Amrita Vishwa Vidhyappetham, India and have submitted a fully automatic run. For this they have first represented the set of prior and current cases as vectors. To do so, they have used the *Doc2Vec* algorithm as implemented in the *gensim* package of python. Once the vectors are obtained the similarity between a query case document and a prior case is simply calculated as the cosine similarity between the two vectors. The top ranked prior cases are reported for each of the current cases.
- **christfire_2017:** This team participated from Christ University, Bangalore, India. They submitted three runs in total and all were fully automatic in nature. The three runs are as described below:
  - **run_1:** The following steps are followed.
    1. Data cleaning and citation context retrieval
    2. Linguistic Preprocessing and creation of Document Term Matrix
    3. Application of Latent Dirichlet Allocation,LDA
    4. Similarity Calculation
  - **run_2:** The following steps are followed:
    1. Data cleaning and citation context retrieval
    2. Linguistic Preprocessing and creation of Document term Matrix
    3. Application of Latent Semantic Analysis, LSA
    4. Similarity Calculation
  - **run_3:** The following steps were followed:
    1. Data cleaning and citation context retrieval
    2. Retaining only nouns from the data
    3. Linguistic Preprocessing and creation of Document term Matrix
    4. Application of Latent Semantic Analysis, LSA to get semantic relationships of nouns
    5. Similarity Calculation

    In the preprocessing part the following steps were followed:
    Case Conversion, Special Character Removal, Number Removal, Stopword Removal, Legal Stopword Removal (Words that appear commonly in all judgments), and Document Stemming.
    The citation context retrieval deals with retaining only those parts of the document that are around the citation markers. The similarity calculation is done by measuring the cosine similarity among the two document vectors (one of the current case another of the prior case). Only the top 50 of the prior cases are reported.

5 RESULTS

Table 2 compares the different runs for Task 1. *RAKE* being the only unsupervised methods has scored significantly lower than other supervised methods. Although CRF with POS and NER performs well it is to be noted that their overall recall is not very good. Whereas,the method using Doc2Vec gives better overall recall.

In Table 3, we have different runs of Task 2 and their evaluation scores. It is to be noted that, using *citation context*(text around the citation markers in the query case), greatly improves performance for the top three methods. Other mentionable well performers would be Dirichlet Prior Smoothing and, TF-IDF vectors over nouns and verbs. These, if used in conjunction with citation context, might as well perform better.

Although, the runs are sorted according to their MAP scores, it is to be noted that, in the legal context the *Overall Recall* is of special importance. As, in real-life, legal practitioners might even consider going through a hundred documents rather than going through just ten of them while missing out some important potential citations. So, a good evaluation technique would be a combination of MAP and Overall Recall.

6 CONCLUDING DISCUSSION

The FIRE 2017 IRLeD track has successfully created a benchmark collection of Legal Case Statements and their Catchphrases by which we can compare the performances of various Catchphrase extraction methods over the legal domain. Also it has created a benchmark citation graph which can be used to evaluate methods for the prior case retrieval tasks. It can be noted that the highest MAP score is **0.390** in Table 3, which reveals the challenge in prior case retrieval.

In future, we plan to conduct other tracks as well, where the following can be considered: (i) Adding supervision to the precedence retrieval task, e.g., by providing a citation network for the documents in the set of prior cases, and (ii) adding new tasks such as document clustering/classification.

ACKNOWLEDGEMENTS

The track organizers thank all the participants for their interest in this track. We also thank the FIRE 2017 organizers for their support in organizing the track.

Run_id	**MAP**	**MRR**	**Prec@10**	**Rec@100**	**Method Summary**
flt_ielab_idf	0.390	0.719	0.236	0.781	IDF, citation context
flt_ielab_plm	0.386	0.710	0.237	0.771	Parsimonious language model, citation context
flt_ielab_para	0.364	0.702	0.221	0.749	Citation context
HLJIT2017_IRLeD_Task2_1	0.329	0.633	0.218	0.681	Dirichlet Prior Smoothing [14]
SSN_NLP_2	0.268	0.546	0.178	0.669	TF-IDF(nouns+verbs)
SSN_NLP_1	0.263	0.518	0.180	0.681	TF-IDF(nouns)
HLJIT2017_IRLeD_Task2_3	0.248	0.525	0.167	0.671	*lucene*
rightstepspune_1_task2	0.202	0.451	0.135	0.564	RegEx, LDA, Doc2Vec
HLJIT2017_IRLeD_Task2_2	0.178	0.407	0.129	0.595	BM25
UB_Botswana_Legal_Task2_R3	0.167	0.348	0.123	0.559	DPH-DFR [2], BoI model[1]
UB_Botswana_Legal_Task2_R1	0.149	0.351	0.112	0.546	DPH-DFR [2]
UB_Botswana_Legal_Task2_R2	0.108	0.302	0.079	0.43	DPH-DFR [2], *Sequential Dependence*[9, 15]
SSN_NLP_3	0.101	0.277	0.076	0.435	Word2Vec(nouns+verbs)
UBIRLeD_2	0.098	0.190	0.069	0.380	LDA
UBIRLeD_3	0.090	0.170	0.062	0.373	LDA
UBIRLeD_1	0.072	0.142	0.049	0.299	BM25
bphcTASK2IRLeD	0.071	0.198	0.060	0.280	POS tags, TF, IDF
AMRITA_CEN_NLP_RBG1_1	0.006	0.015	0.003	0.058	Doc2Vec
christfire_2017_3	0.005	0.011	0.003	0.033	LSA(nouns only)
christfire_2017_2	0.003	0.010	0.002	0.044	LSA
christfire_2017_1	0.002	0.006	0.003	0.016	LDA

Table 3: Evaluation of runs for Task 2: Precedence Retrieval. Runs are sorted in descending order of the MAP or Mean average Precision.

REFERENCES

[1] G. Amati. 2003. Probabilistic Models for Information Retrieval based on Divergence from Randomness. *University of Glasgow,UK, PhD Thesis* (June 2003), 1 – 198.

[2] G. Amati, E. Ambrosi, M. Bianchi, C. Gaibisso, and G. Gambosi. 2007. FUB, IASI-CNR and University of Tor Vergata at TREC 2007 Blog Track. In *Proceedings of the 16th Text REtrieval Conference (TREC-2007)*. Text REtrieval Conference (TREC), Gaithersburg, Md., USA., 1–10.

[3] Leo Breiman. 2001. Random Forests. *Mach. Learn.* 45, 1 (Oct. 2001), 5–32. DOI: http://dx.doi.org/10.1023/A:1010933404324

[4] Stefanie Brüninghaus and Kevin D. Ashley. 2001. Improving the Representation of Legal Case Texts with Information Extraction Methods. In *Proceedings of the 8th International Conference on Artificial Intelligence and Law (ICAIL '01)*. ACM, New York, NY, USA, 42–51. DOI:http://dx.doi.org/10.1145/383535.383540

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

[6] Arpan Mandal, Kripabandhu Ghosh, Arindam Pal, and Saptarshi Ghosh. 2017. Automatic Catchphrase Identification from Legal Court Case Documents. In *Proc. ACM Conference on Information and Knowledge Management (CIKM)*. 2267–2270.

[7] Michael McCandless, Erik Hatcher, and Otis Gospodnetic. 2010. *Lucene in Action, Second Edition: Covers Apache Lucene 3.0.* Manning Publications Co., Greenwich, CT, USA.

[8] Olena Medelyan. 2009. Human-competitive automatic topic indexing. (2009). http://cds.cern.ch/record/1198029 Presented on July 2009.

[9] Donald Metzler and W. Bruce Croft. 2005. A Markov Random Field Model for Term Dependencies. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '05)*. ACM, New York, NY, USA, 472–479. DOI:http://dx.doi.org/10.1145/1076034.1076115

[10] J.L.T. Olsson. 1999. Guide To Uniform Production of Judgments, 2nd edn. *Australian Institute of Judicial Administration, Carlton South* (1999).

[11] J. R. Quinlan. 1986. Induction of decision trees. *Machine Learning* 1, 1 (01 Mar 1986), 81–106. DOI:http://dx.doi.org/10.1007/BF00116251

[12] Stephen Robertson and Hugo Zaragoza. 2009. The Probabilistic Relevance Framework: BM25 and Beyond. *Foundations and Trends in Information Retrieval* 3, 4 (April 2009), 333–389.

[13] Stuart Rose, Dave Engel, Nick Cramer, and Wendy Cowley. 2010. *Automatic Keyword Extraction from Individual Documents*. John Wiley and Sons, Ltd, 1–20. DOI:http://dx.doi.org/10.1002/9780470689646.ch1

[14] Fei Song and W. Bruce Croft. 1999. A General Language Model for Information Retrieval. In *Proceedings of the Eighth International Conference on Information and Knowledge Management (CIKM '99)*. ACM, New York, NY, USA, 316–321. DOI:http://dx.doi.org/10.1145/319950.320022

[15] Edwin Thuma, Nkwebi Peace Motlogelwa, and Tebo Leburu-Dingalo. 2017. UB-Botswana Participation to CLEF eHealth IR Challenge 2017: Task 3 (IRTask1 : Ad-hoc Search). In *Working Notes of CLEF 2017 - Conference and Labs of the Evaluation Forum, Dublin, Ireland, September 11-14, 2017.* http://ceur-ws.org/Vol-1866/paper_73.pdf