Proceedings of the Sixth International Competition on Legal Information Extraction/Entailment (COLIEE 2019)

in association with
the 17th International Conference
on Artificial Intelligence and Law

COLIEE 2019 Organizers

Randy Goebel
Yoshinobu Kano
Mi-Young Kim
Juliano Rabelo
Ken Satoh
Masaharu Yoshioka

University of Montreal, Montreal, Canada
June 21, 2019

Preface

This volume contains the papers accepted for presentation at COLIEE 2019, the Sixth Competition on Legal Information Extraction/Entailment, held on June 21, 2019 in Montréal, Canada, in conjunction with ICAIL 2019, the 17th International Conference on Artificial Intelligence and Law. Papers for COLIEE are focused on an international legal information processing competition, which includes competitions on automatically understanding both statute law and case law. This year there were 18 submissions. Each submission was reviewed by at least 2, and on the average 2.9, program committee members. The committee decided to accept 3 winnig papers for each task at the ICAIL main conference and to accept 14 papers for the workshop.

The COLIEE workshop program will also include 1 invited talk. The COLIEE competition was originally proposed by Ken Satoh from the National Institute of Informatics in Tokyo, driven by his experience as a computer science professor in law school, and his experience with the challenge of Japanese Bar examinations. The idea was to formulate a challenging legal informatics competition that would engage researchers around the world, and build a community that would consider all manner of computer science and Artificial Intelligence methods to tackle the problems of legal reasoning.

Now after just completing the sixth year of competitions, Ken's goal has been achieved. We have had six competitions, all of whose selected contributions have been peer-reviewed, published at ICAIL proceedings or at the Springer Lecture Notes in Artificial Intelligence series, and – most importantly – developed a community of researchers, lawyers, judges, and related communities to discuss the impact and future of technology adoption in for legal systems. The team around Ken now includes Yoshinobu Kano from Shizuoka University, Masaharu Yoshioka from Hokkaido University, and the team from the Alberta Machine Intelligence Institute including Mi-Young Kim, Juliano Rabelo, and Randy Goebel. As our community has developed, the legal informatics industry has also engaged, and we are most grateful for the support from Jimoh Ovbiagele and Andrew Arruda from Ross Intelligence Young-Yik Rhim from Intellicon (Seoul), and Colin LaChance from VLex. As all of know, the Easychair team is constantly responding to extend their conference management system in ways that helps us accomplish what would otherwise be an impossible task for all us volunteers.

June 21, 2019
University of Montreal, Montreal, Canada

Randy Goebel
Yoshinobu Kano
Mi-Young Kim
Juliano Rabelo
Ken Satoh
Masaharu Yoshioka
COLIEE 2019 organizers

Additional Reviewers

Le Minh, Nguyen
Tran, Vu

Table of Contents

COLIEE 2019 Overview

Juliano Rabelo
Alberta Machine Intelligence Institute
University of Alberta
Edmonton, AB, Canada
rabelo@ualberta.ca

Mi-Young Kim
Department of Science, Augustana Faculty
University of Alberta
Camrose, AB, Canada
miyoung2@ualberta.ca

Randy Goebel
Alberta Machine Intelligence Institute
University of Alberta
Edmonton, AB, Canada
rgoebel@ualberta.ca

Masaharu Yoshioka
Graduate School of Information Science and Technology
Global Station for Big Date and Cybersecurity, Global Institution for Collaborative Research and Education
Hokkaido University
Kita-ku, Sapporo-shi, Hokkaido
Japan
yoshioka@ist.hokudai.ac.jp

Yoshinobu Kano
Faculty of Informatics
Shizuoka University
Naka-ku, Hamamatsu-shi, Shizuoka
Japan
kano@inf.shizuoka.ac.jp
nkiyota@kanolab.net

Ken Satoh
National Institute of Informatics
Hitotsubashi, Chiyoda-ku, Tokyo
Japan
ksatoh@nii.ac.jp

ABSTRACT

In this paper, we summarize the evaluation of the 6th Competition on Legal Information Extraction/Entailment (COLIEE 2019). Four tasks are promoted in the Competition with a focus on case law and statute law. The case law component includes an information retrieval task (Task 1), and the confirmation of an entailment relation between an existing case and an unseen case (Task 2). The statute law component includes an information retrieval task (Task 3) and an entailment/question answering task (Task 4). Participation was open to any group based on any approach. 11 different teams participated in the case law competition tasks, some of them in more than one task. We received results from 7 teams for Task 1 (15 runs) and 7 teams for Task 2 (18 runs). Regarding the statute law, there were 8 different teams participating, some in more than one task. 7 teams submitted a total of 13 runs for Task 3, and 7 teams submitted a total of 15 runs for Task 4. We describe each team's approaches, our official evaluation, and analysis on our data and submission results.

KEYWORDS

legal documents processing, textual entailment, information retrieval, classification, question answering

ACM Reference Format:
Juliano Rabelo, Mi-Young Kim, Randy Goebel, Masaharu Yoshioka, Yoshinobu Kano, and Ken Satoh. 2019. COLIEE 2019 Overview. In *COLIEE '19: Competition on Legal Information Extraction/Entailment, June 21, 2019, Montreal, Canada.* ACM, New York, NY, USA, 9 pages.

COLIEE 2019, June 21, 2019, Montreal, Quebec

1 INTRODUCTION

The Competition on Legal Information Extraction/Entailment (COLIEE) is a series of evaluation competitions to develop the state of the art for information retrieval and entailment using legal texts. It is usually co-located with JURISIN, the Juris-Informatics workshop series, which was created to promote community discussion on both fundamental and practical issues on legal information processing, with the intention to embrace various disciplines, including law, social sciences, information processing, logic and philosophy, including the existing conventional "AI and law" area. In COLIEE editions 2014 to 2017, there were two tasks (information retrieval (IR) and entailment) using Japanese Statue Law (civil law). Since COLIEE 2018, two new tasks (IR and entailment) were introduced which use Canadian case law (Tasks 1 and 2).

Task 1 is a legal case retrieval task, and it involves reading a new case Q, and extracting supporting cases S1, S2, ..., Sn from the provided case law corpus, hypothesized to support the decision for Q. Task 2 is the legal case entailment task, which involves the identification of a paragraph or paragraphs from existing cases, which entail a given fragment of a new case. For the information retrieval task (Task 3), based on the discussion about the analysis of previous COLIEE IR tasks, we modify the evaluation measure of the final results and also ask the participants to submit ranked relevant articles results to discuss the detailed difficulty of the questions. For the entailment task (Task 4), we performed categorized analyses to show different issues of the problems and characteristics of the submissions, in addition to the evaluation accuracy as in previous COLIEE tasks.

The rest of the paper is organized as follows: Sections 2, 3, 4, 5 decribe each task, presenting their definitions, datasets, list of approaches submitted by the participants, and results attained. Section 6 presents final some final remarks.

2 TASK 1 - CASE LAW INFORMATION RETRIEVAL

2.1 Task Definition

This task consists in finding which cases, in the set of candidate cases, should be "noticed" with respect to a given query case. "Notice" is a legal technical term that denotes a legal case description that is considered to be relevant to a query case. More formally, given a query case q and a set of candidate cases $C = \{c_1, c_2, ..., c_n\}$, the task is to find the supporting cases $S = \{s_1, s_2, ..., s_n \mid s_i \in C \wedge noticed(s_i, q)\}$ where $noticed(s_i, q)$ denotes a relationship which is true when $s_i \in S$ is a noticed case with respect to q.

2.2 Dataset

The training dataset consists of 285 base cases, each with 200 candidate cases from which the participants must identify those that should be noticed with respect to the base case. The golden labels for this dataset were disclosed upfront to this task's participants. The official COLIEE test dataset has 61 cases and was initially released without the golden labels, which were only disclosed after the competition results were published. Table 1 summarizes the properties of those datasets.

2.3 Approaches

Seven teams submitted a total of 15 runs for this task. Deep learning techniques and machine learning based classifiers were commonly used. More details on the approaches are show below:

- **CACJ (one run)** [3] applies a machine learning based classifier using features extracted from the cases header (i.e., it does not consider any of the case contents).
- **CLArg (one run)** [17] describes an approach based on vector representation of cases, in combination with two different classifiers: random forests and k-nearest neighbours. Additionally, the authors notice the inclusion of a feature consisting of how many times a particular past case has appeared in the entire dataset augments their results.
- **HUKB (one run)** [26] improved their previous system, used on the 2018 COLIEE edition (which was based on the use of structural information which considers a case as composed of three sections: header, facts and footer), by incorporating the use of case metadata: date, to exclude candidates more recent than the base case, and topics.
- **IITP (three runs)** [4] uses a combination of Deep Learning techniques, such as Doc2Vec, and Information Retrieval techniques, such as BM25, to tackle the task 1 challenge.
- **ILPS (three runs)** [21] combines text summarizing and a generalized language model (BERT) in order to assess pairwise relevance. To overcome a limitation of the framework on handling text fragments longer than 512 tokens, the authors apply summarization techniques over the case contents. The generated embeddings are then fed to an MLP classifier.
- **JNLP (three runs)** [23] apply a summarization model which encodes a document into a continuous vector space which embeds the summary properties of the document. The authors combine such encoded representation with latent and lexical features extracted from different parts of a given query and its candidates.
- **UA (three runs)** [19] developed an approach based on the use of the Universal Sentence Encoder to generate a vector representation of both the base case and each candidate, and then calculation of a similarity score through the cosine measure (this approach was used as the baseline for this task).

2.4 Results

The F1-measure is used to assess performance in this task. We use a simple baseline model thath uses the Universal Sentence Encoder to encode each candidate case and base case into a fixed size vector, and then applies the cosine distance between both vectors. The baseline result was 0.3560 (precision: 0.3333, recall: 0.3443, for a threshold of 0.57 minimum similarity). The actual results of the submitted runs by all participants are shown on table 2, from which it can be seen that only 1 team could not reach the baseline. The submissions from the UA team was exactly based on this baseline method.

Table 2 shows JNLP attained the best result for the F1-score, the official metric used in this task. CLArg had the best score when only precision is considered, whereas CACJ had the best recall score. The F1-score for CLArg and CACJ, however, were not among the best ones for this task, which shows the difficulty of finding the right balance in the system to achieve a good overall performance in this kind of problem.

3 TASK 2 - CASE LAW ENTAILMENT

3.1 Task Definition

Given a base case and a specific fragment from it, and a second case which is relevant in respect to the base case, this task consists in determining which paragraphs of the second case entail that fragment of the base case. More formally, given a base case b and its entailed fragment f, and another case r represented by its paragraphs $P = \{p_1, p_2, ..., p_n\}$ such that $noticed(b, r)$ as defined in section 2 is true, the task consists in finding the set $E = \{p_1, p2, ..., p_m \mid p_i \in P\}$ where $entails(p_i, f)$ denotes a relationship which is true when $p_i \in P$ entails the fragment f.

3.2 Dataset

The training dataset has 181 base cases, each with its respective entailed fragment in a separate file. For each base case, a related case represented by a list of paragraphs is given, from which the paragraph(s) that entail the base-case-entailed fragment must be identified. The golden labels for this dataset were disclosed upfront to this task's participants. The official COLIEE test dataset has 44 cases and was initially released without the golden labels, which were only disclosed after the competition results were published. Table 3 summarizes the properties of those datasets.

3.3 Approaches

Seven teams submitted a total of 18 runs to this task. The most used techniques were those based on transformer methods, such as

Table 1: Summary for the Case Law Retrieval Task Datasets

Property	Training	Testing
Number of base cases	285	61
Total number of candidate cases	57,000	12,200
Total number of noticed cases	1486 (2.60%)	330 (2.70%)

Table 2: Results attained by all teams on the test dataset of task 1.

Team	Submission File	Precision	Recall	F1-score
JNLP	JNLP.task_1.pl.txt	0.6000	0.5545	0.5764
JNLP	JNLP.task_1.ple.txt	0.6000	0.5545	0.5764
JNLP	JNLP.task_1.p.txt	0.5934	0.5485	0.5701
ILPS	BERT_Score_0.946.txt	0.6810	0.4333	0.5296
HUKB	task1.HUKB	0.7021	0.4000	0.5097
ILPS	BM25_Rank_6.txt	0.4672	0.5182	0.4914
ILPS	BERT_Score_0.96.txt	0.8188	0.3424	0.4829
IITP	task1.IITPdocBM.txt	0.6368	0.3879	0.4821
IITP	task1.IITPBM25.txt	0.6256	0.3848	0.4765
CLArg	CLarg.txt	0.9266	0.3061	0.4601
IITP	task1.IITPd2v.txt	0.4653	0.3455	0.3965
UA	UA_0.57.txt	0.3560	0.3333	0.3443
UA	UA_0.52.txt	0.3513	0.3364	0.3437
UA	UA_0.54.txt	0.3639	0.3242	0.3429
CACJ	submit_task1_CACJ01.csv	0.2119	0.5848	0.3110

Table 3: Summary for the Case Law Entailment Task Datasets

Property	Training	Testing
Number of base cases	181	44
Total paragraphs in the related cases	5,814	1,448
Total true entailing paragraphs	202 (3.47%)	45 (3.10%)

BERT [2] or ELMo [18]. More details on the approaches are show below.

- **IeLab** [1] (three runs)] used an IR-based technique which selects terms from the entailed fragments and the candidates using inverse document frequency and part of speech information.
- **IITP (three runs)** [4] describes an approach which uses BM25, an Information Retrieval technique, and Doc2Vec, a Deep Learning based technique, for this task.
- **JNLP (three runs)** has not submitted a paper describing the details of their approach for task 2, but they devised deep learning based methods for other tasks of COLIEE 2019 (e.g., [16]).
- **TRCase (one run)** [13] applies a ranking algorithm which uses word embeddings and textual similarity features to determine entailment relationships between a candidate paragraph and an entailed fragment. The authors observe that the set of selected features provide better results when applied to a ranking approach, rather than a supervised classifier.
- **TTCL (three runs)** presents an approach based on a generalized language model using BERT for the case law entailment task, comparing that approach with an SVM baseline approach. To overcome the framework limitation of 512 tokens, the authors apply BERT at a sentence level, considering a paragraph to be an entailing example when one or more sentences are classified as entailing one or more sentences from the entailed fragment.
- **UA (three runs)** [19] proposes an approach which relies the extraction of similarity measures between the candidate paragraph and the entailed fragment; the application of BERT on those two pieces of text; use of a threshold-based classifier; and post-processing the results considering the a priori probability determined by the data distribution on the training samples.
- **UBLTM (two runs)** has not submitted a paper describing the details of their approach.

[1] This is an interesting approach worth further investigation, however the paper describing the method lacks important information and thus was not accepted for publication on the COLIEE proceedings

3.4 Results

The F1-measure is used to assess performance in this task. The score attained by a simple baseline model which uses the Universal Sentence Encoder to encode each candidate paragraph and the entailed fragment into a fixed size vector and applies the cosine distance between both vectors was 0.1760 (precision: 0.1375, recall: 0.2444, for a threshold of 0.75 minimum similarity). The actual results of the submitted runs by all participants are shown on table 4, from which it can be seen that only 2 runs had a performance worse than the baseline score (however, the teams which sent those submissions also got better results on other runs).

From Table 4, one can see UA attained the best result for the F1-score, the official metric used in this task. However, IITP and TRCase achieved comparable results for the F1-score. It is also worth noting that IITP attained the best score considering only precision, and TTCL got the best recall score.

4 TASK 3 - STATUTE LAW INFORMATION RETRIEVAL

4.1 Task Definition

This task involves reading a legal bar exam question Q, and identification of a subset of Japanese Civil Code Articles S_1, S_2,..., S_n from the entire Civil Code which are those appropriate for answering the question such that

$$Entails(S_1, S_2, ..., S_n, Q) \text{ or } Entails(S_1, S_2, ..., S_n, notQ).$$

Given a question Q and the all Civil Code Articles, the participants are required to retrieve the set of "$S_1, S_2, ..., S_n$" as the answer of this track.

4.2 Dataset

For task 3, questions related to Japanese civil law were selected from the Japanese bar exam. The organizers provided a data set used for previous bar law exams, translated to English [8–10, 25] as training data (717 questions), with new questions selected from the 2018 bar exam as test data (98 questions).

The number of questions classified by the number of relevant articles is listed in Table 5.

4.3 Approaches

The following seven teams submitted their results (13 runs in total). Four teams (HUKB, JNLP, KIS and UA) had experience in submitting results in the previous competition, and three teams (DBSE, EVORA and IITP) were new competitors. Common techniques used in the system were well known IR engine mechanisms such as elasticsearch [2], Terrier [12], Indri [22], gensim [3], scikit-learn [4] with various scoring function such as TF-IDF, BM25. For the indexing, the most common method was ordinal word base indexing with stemming. Several teams use N-gram, word sequence, Word2Vec [15] and Doc2Vec[11].

- **DBSE (one run)** [24] used BM25 scoring of elasticsearch and Word2Vec [15] based similarity scoring. They finally select the one or more results from them.
- **EVORA (three runs)** [20] uses Terrier IR platform with different scoring function with two query sets (original and keyword selection) and two article database (original articles and keyword selected articles).
- **HUKB (one run)** [26] uses sentence structure analysis to extract condition part and argument part of the query and articles and compare the similarity using Indri IR system. Final results are calculated by SVMRank using those features.
- **KIS (two runs)** [5] uses Doc2Vec [11] for generating document embedding vector and calculate similarity among query and articles. They also use TF-IDF to select important keywords for generating document embedding. Final results are selected by considering the score difference between the top ranked and candidate documents.
- **IITP (two runs)** [4] uses BM25 module of gensim and tfidf module of scikit learn.
- **JNLP (two runs)** [1] proposed to use different indexing method for TF-IDF calculation (N-gram, verb-phrase, noun-phrase) and calculate similarity using cosine similarity. Final results are selected by considering the score difference between the i-th ranked document and i + 1-th ranked one.
- **UA (two runs)** [14] uses the TF-IDF model and language model as an IR module.

The teams which participated in the previous COLIEE proposed an extension or equivalent system for Task 3, and new teams proposed methods that were different from those previous.

4.4 Results

Table 6 shows the evaluation results of submitted runs. The official evaluation measures used in this task were macro average (average of evaluation measure values for each query over all queries) of F2 measure, precision (Prec.), and recall (Rec.). The terms "ret.", and "rel" represent the number of returned articles and the number of returned relevant articles, respectively.

$$precision = \frac{number\ of\ retrieved\ relevant\ articles}{number\ of\ returned\ articles} \quad (1)$$

$$recall = \frac{number\ of\ retrieved\ relevant\ articles}{number\ of\ relevant\ articles} \quad (2)$$

$$f2 = \frac{5 \times precision \times recall}{4 \times precision + recall} \quad (3)$$

We also calculate the mean average precision (MAP), recall at k (R_k: recall calculated by using the top k ranked documents as returned documents) by using the long ranking list (100 articles). Table 6 shows

From the official evaluation measure F2, UA-TFIDF is the best run among all submitted runs. Since KIS_2 returns large numbers of candidate articles, recall of KIS_2 is the best. For the longer ranked list, EVORA is better than others.

Figure 1,2,3 shows average of evaluation measure for all submission runs. As we can see from Figure 1, there are many easy questions that almost all system can retrieve the relevant article.

[2] https://www.elastic.co/
[3] https://radimrehurek.com/gensim/
[4] https://scikit-learn.org

Table 4: Results attained by all teams on the test dataset of task 2.

Team	Submission File	Precision	Recall	F1-score
UA	UA_0.400000.txt	0.6538	0.7556	0.7010
UA	UA_0.250000.txt	0.6364	0.7778	0.7000
IITP	task2.iitpBM25.txt	0.7045	0.6889	0.6966
UA	UA_0.300000.txt	0.6296	0.7556	0.6869
TRCase	TRCase_colie_test_submission_task2	0.6818	0.6667	0.6742
IITP	task2.iitp2docBM.txt	0.6591	0.6444	0.6517
JNLP	JNLP.task_2.lex.txt	0.5909	0.5778	0.5843
TTCL	uncased758256.txt	0.4000	0.8000	0.5333
TTCL	uncased758voted.txt	0.3882	0.7333	0.5077
TTCL	uncased758512.txt	0.3780	0.6889	0.4882
ielab	ielabsen.txt	0.4545	0.4444	0.4494
ielab	ielabphrase.txt	0.3409	0.3333	0.3371
ielab	ielabterm.txt	0.2273	0.2222	0.2247
UBLTM	UBLTM_T2_2.txt	0.1273	0.6222	0.2113
UBLTM	UBLTM_T2_1.txt	0.1182	0.5778	0.1962
JNLP	JNLP.task_2.cls-elmo.txt	0.1364	0.1333	0.1348
JNLP	JNLP.task_2.cls-elmobert.txt	0.0682	0.0667	0.0674
IITP	task2.iitp2D2v.txt	0.0455	0.0444	0.0449

Table 5: Number of questions classified by number of relevant articles

number of relevant article(s)	1	2	3	5	total
number of questions	80	15	2	1	98

Table 6: Evaluation results of submitted runs (Task 3) and the corresponding organizers' run

runid	lang	ret.	rel.	F2	Prec.	Rec.	MAP	R_5	R_{10}	R_{30}
DBSE	E	172	54	0.466	0.454	0.493	0.512	0.512	0.620	0.669
EVORA1	E	98	56	0.533	0.571	0.529	**0.628**	**0.669**	0.744	**0.851**
EVORA2	E	98	56	0.533	0.571	0.529	0.617	0.653	0.744	0.835
EVORA3	E	98	56	0.529	0.571	0.524	0.624	0.653	**0.752**	0.835
iitpBM25	E	98	48	0.447	0.490	0.442	0.541	0.620	0.669	0.760
iitptfidf	E	98	43	0.401	0.439	0.396	0.506	0.570	0.628	0.752
JNLP-tf	E	165	64	0.534	0.459	0.582	0.598	0.653	0.686	0.769
JNLP-tfnv	E	171	61	0.505	0.403	0.562	0.575	0.595	0.653	0.769
UA-LM	E	98	48	0.452	0.490	0.447	0.541	0.554	0.636	0.727
UA-TFIDF	E	98	58	**0.549**	**0.592**	0.544	0.618	0.620	0.694	0.760
HUKB	J	98	44	0.414	0.449	0.410	0.494	0.488	0.612	0.727
KIS	J	404	69	0.503	0.423	0.613	0.562	0.628	0.711	0.835
KIS_2	J	408	**72**	0.503	0.427	**0.637**	0.540	0.653	0.744	0.835

However, there are also many queries for which none of the system can retrieve the relevant articles.

One of the example of this question is H30-1-A: "An unborn child may not be given a gift on the donor's death." and relevant article is Article 3 "The enjoyment of private rights shall commence at birth." There is no common words between the query and a relevant article and it requires knowledge about relationship between "commence at birth" and "unborn" for understanding the relationship. In order to analyze the improvement of the system for such difficult questions, it is necessary to compare the retrieval performance for such difficult queries.

5 TASK 4 - STATUTE LAW ENTAILMENT

5.1 Task Definition

Task 4 is a task to determine entailment relationships between a given problem sentences and article sentences. Competitor systems should answer "yes" or "no" regarding the given problem sentences and given article sentences. Until COLIEE 2016, the competition had pure entailment tasks, where t1 (relevant article sentences)

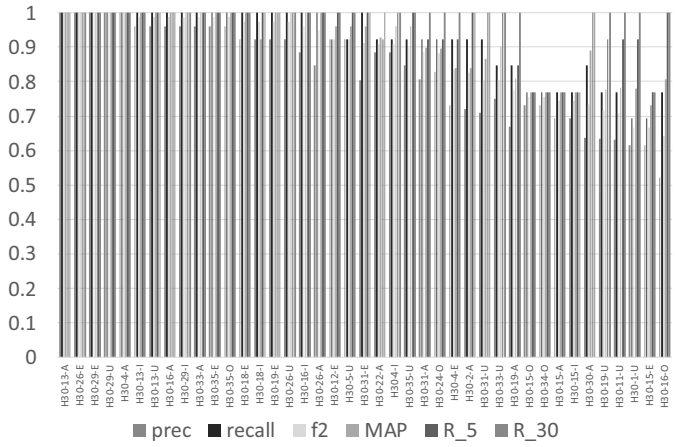


Figure 1: Averages of precision, recall, F2, MAP, R_5, and R_30 for easy questions with a single relevant article

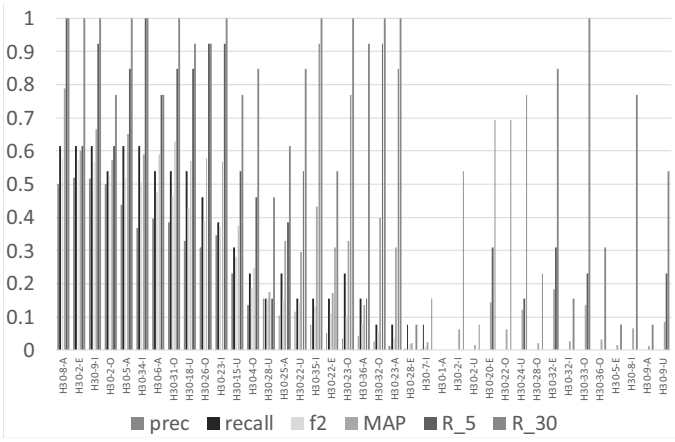


Figure 2: Averages of precision, recall, F2, MAP, R_5, and R_30 for noneasy questions with a single relevant article

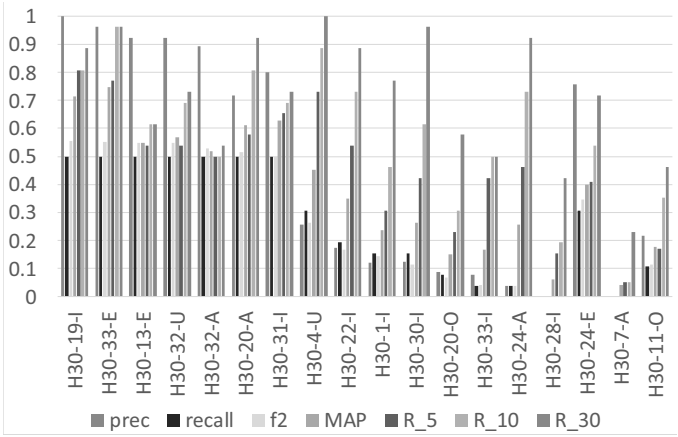


Figure 3: Averages of precision, recall, F2, MAP, R_5, R_10, and R_30 for noneasy questions with a single relevant article

and t2 (problem sentence) were given. Due to the limited number of available problems, COLIEE 2017 and 2018 did not retain this style of task. In the Task 4 of COLIEE 2019, we returned to the pure textual entailment task to attract more participants, allowing more focused analyses.

5.2 Dataset

Our training dataset and test dataset are the same as Task 3. Questions related to Japanese civil law were selected from the Japanese bar exam. The organizers provided a data set used for previous campaigns as training data (717 questions) and new questions selected from the 2018 bar exam as test data (98 questions).

5.3 Approaches

The following seven teams submitted their results (15 runs in total). Two teams (KIS and UA) had experience in submitting results in the previous campaign. We describe each system's overview below.

- **UA [14]** uses condition/conclusion/exception detection rules, and negation dictionaries created manually. They translated original Japanese texts into Korean by machine translation, employed their own Korean parser and Korean resources. **UA_Ex** uses Excite machine translation service, **UA_Go** uses Google machine translation service. Their system is same as in the previous COLIEE task.
- **KIS [6]** parses sentences into predicate-argument structures to compare t1/t2 pairs, detecting negations and conditions. They use an ensemble of different comparison criteria (**KIS_3module**), then adding their own synonym dictionary (**KIS_dic**) or using FrameNet (**KIS_frame**).
- **IITP** [4] uses BERT with a BERT-base model.
- **DBSE** [24] uses an ensemble of stacked LSTMs.
- **JNLP** [16] indirectly solves the original problem with a derived problem with more abundant data. They trained using a stacked GRU.
- **TR** [7] uses BERT large model with decomposable attention (**TRAttn**), and similarity features (**TRSimFeat**).
- **EVORA** [20] used deep neural networks based methods, such as embedding by FastText (**EVORA1**), LSTM (**EVORA2**) and CNN (**EVORA3**).

5.4 Results

Evaluation was performed by accuracy. Table 7 shows evaluation results of Task 4 for each submitted run.

Because an entailment task is essentially a complex composition of different subtasks, we manually categorized our test data into categories, depending on what sort of technical issues are required to be resolved. Tables 8 and 9 show our categorization results. As this is a composition task, overlap is allowed between categories. Our categorization is based on the original Japanese version of the legal bar exam.

Although some cells show better results than others, none of the current systems could have solved problem types of higher semantics, e.g. anaphora resolutions. We need more precise survey what made the difference between the systems, especially which part is firmly solved to output stable correct results.

6 FINAL REMARKS

We have summarized the results of the COLIEE 2019 edition. On case law, Task 1 deals with the retrieval of noticed cases, and Task 2 poses the problem of identifying which paragraphs of a relevant case entail a given fragment of a new case. On statute law, Task 3 is about retrieving articles to decide the appropriateness of the legal question, and Task 4 is a task to entail whether the legal question is correct or not. 11 different teams participated in the case law competition (some of them in both tasks). We received results from 7 teams for Task 1 (a total of 15 runs), and 7 teams for Task 2 (a total of 18 runs). Regarding the statute law tasks, there were 8 different teams participating, some in both tasks. 7 teams submitted 13 runs for Task 3, and 7 teams submitted 15 runs for Task 4.

A variety of methods were used for Task 1: classification using only features extracted from the case header, random forest and k-NN classifiers, exploitation of the case structure information, deep learning based techniques (such as transformer methods and tools such as the Universal Sentence Encoder), lexical and latent features, embedding summary properties, and information retrieval techniques were the main ones. For Task 2, transformer-based tools such as BERT and ELMo were prevalent, but IR techniques and textual similarity features have also been applied. The results attained were satisfactory, but there is much room for improvement, especially if one considers the related issue of explaining the predictions made; deep learning methods, which showed promising results this year, would not be so appropriate in a scenario where explainability is key. For future editions of COLIEE, we plan on continuing expanding the data sets in order to improve the robustness of results, as well as evaluating ways of introducing explainability-aware tasks or requirements into the competition.

For Task 3, we found there are three types of questions in the test data (easy questions, difficult questions with vocabulary mismatch, and questions with multiple answers). Most of the submission systems are good at retrieving relevant answers for easy questions, but it is still difficult to retrieve relevant articles for other question types. It may be necessary to focus on such question types to improve the overall performance of the IR system. For Task 4, overall performance of the submissions is still not sufficient to use their systems in real applications, mainly due to lack of coverage for some classes of problems, such as anaphora resolution. We found this task is still a challenging one to discuss and develop deep semantic analysis issues in the real application and natural language processing in general.

ACKNOWLEDGMENTS

This research was supported by the National Institute of Informatics, Shizuoka University, Hokkaido University, and the Alberta Machine Intelligence Institute (AMII). Thanks to Colin Lachance from vLex for his constant support in the development of the case law data set, and to support from Ross Intelligence and Intellicon.

REFERENCES

[1] Tran-Binh Dang, Thao Nguyen, and Le-Minh Nguyen. 2019. An approach to Statute Law Retrieval Task in COLIEE-2019. In *Proceedings of the 6th Competition on Legal Information Extraction/Entailment (COLIEE'2019).*

[2] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *CoRR* abs/1810.04805 (2018). arXiv:1810.04805 http://arxiv.org/abs/1810.04805

Table 7: Evaluation results of submitted runs (Task 4)

Team	Dataset Language	# of correct answers (98 problems in total)	Accuracy
UA_Ex	Japanese	67	0.6837
KIS_3module	Japanese	61	0.6224
IITP	English	58	0.5918
KIS_dic	Japanese	58	0.5918
UA_Go	Japanese	58	0.5918
KIS_frame	Japanese	57	0.5816
DBSE	English	56	0.5714
JNLP.t=98	English	56	0.5714
TRAttn	English	55	0.5612
TRSimFeat	English	52	0.5306
JNLP.t=85	English	51	0.5204
EVORA1	?	50	0.5102
JNLP.t=78	English	48	0.4898
EVORA3	?	47	0.4796
EVORA2	?	44	0.4490

Table 8: Technical category statistics of questions, and correct answers of submitted runs for each category in numbers of counts. Alphabetical letters stand for team names; a: DBSE, b: EVORA1, c: EVORA2, d: EVORA3, e: IITP, f: JNLP: t=78, g: JNLP(t=85), h: JNLP(t=98), i: KIS_3module, j: KIS_dic, l: TRAttn, m: TRSimFeat, n: UA_Ex, o: UA_Go, respectively. Team columns stand for their number of correct answers for corresponding category.

category	#	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
Conditions	83	47	44	**34**	38	49	40	42	46	52	50	47	48	44	**57**	51
Person role	66	36	37	**24**	26	40	34	35	38	43	39	41	35	34	**46**	37
Person Relationship	66	36	37	**24**	26	40	34	35	38	43	39	41	35	34	**46**	37
Negation	44	27	24	20	20	26	22	**19**	26	26	**29**	23	23	22	26	24
Entailment	33	18	14	18	15	16	15	**10**	16	**21**	19	19	18	16	20	18
Dependency	28	12	14	**10**	11	14	**10**	17	16	19	17	18	19	12	**21**	17
Ambiguity	26	11	14	**9**	14	12	10	15	**17**	15	15	11	12	14	**17**	13
Legal terms	24	12	11	11	14	12	**9**	14	**18**	13	12	15	15	13	17	15
Anaphora	22	12	13	**9**	12	13	13	13	13	16	12	14	**9**	**9**	**18**	14
Verb paraphrase	21	11	12	**8**	**8**	**15**	10	10	11	10	13	11	12	10	13	9
Morpheme	18	13	7	**6**	8	13	13	11	9	12	11	13	**14**	12	**14**	**14**
Case role	17	8	10	**6**	8	**12**	7	8	10	9	7	7	11	8	9	8
Predicate argument	14	**5**	7	6	7	10	7	7	8	9	7	9	9	**5**	**11**	7
Article search	11	5	6	4	7	8	7	6	**3**	8	8	5	4	4	**10**	7
Paraphrase	11	4	6	5	7	3	**1**	7	7	9	3	**10**	4	7	8	5
Itemized	8	**6**	4	5	**3**	5	4	**3**	4	4	**6**	4	**6**	5	**6**	5
Normal terms	7	3	3	**2**	**5**	4	**2**	4	4	**5**	**5**	4	3	**2**	**5**	4
Calculation	2	1	1	1	1	1	1	1	1	1	1	1	1	1	2	2

[3] Rajaa El Hamdani, Aurore Troussel, and Claire Houvenagel. 2019. COLIEE Case Law Competition Task 1: The Legal Case Retrieval Task. In *Proceedings of the 6th Competition on Legal Information Extraction/Entailment (COLIEE'2019).*

[4] Baban Gain, Dibyanayan Bandyopadhyay, Tanik Saikh, and Asif Ekbal. 2019. IITP@COLIEE 2019: Legal Information Retrieval using BM25 and BERT. In *Proceedings of the 6th Competition on Legal Information Extraction/Entailment (COLIEE'2019).*

[5] Ryuji Hayashi and Yoshinobu Kano. 2019. Searching Relevant Articles for Legal Bar Exam by Doc2Vec and TF-IDF. In *Proceedings of the 6th Competition on Legal Information Extraction/Entailment (COLIEE'2019).*

[6] Reina Hoshino, Naoki Kiyota, and Yoshinobu Kano. 2019. Question Answering System for Legal Bar Examination using Predicate Argument Structures focusing on Exceptions. In *Proceedings of the 6th Competition on Legal Information Extraction/Entailment (COLIEE'2019).*

[7] John Hudzina, Thomas Vacek, Kanika Madan, Custis Tonya, and Frank Schilder. 2019. Statutory entailment using similarity features and decomposable attention models. In *Proceedings of the 6th Competition on Legal Information Extraction/Entailment (COLIEE'2019).*

[8] Yoshinobu Kano, Mi-Young Kim, Randy Goebel, and Ken Satoh. 2017. Overview of COLIEE 2017. In *COLIEE 2017. 4th Competition on Legal Information Extraction and Entailment (EPiC Series in Computing)*, Ken Satoh, Mi-Young Kim, Yoshinobu Kano, Randy Goebel, and Tiago Oliveira (Eds.), Vol. 47. EasyChair, 1–8. https://doi.org/10.29007/fm8f

[9] Mi-Young Kim, Randy Goebel, Yoshinobu Kano, and Ken Satoh. 2016. COLIEE-2016: evaluation of the competition on legal information extraction and entailment. In *International Workshop on Juris-informatics (JURISIN 2016).*

[10] Mi-Young Kim, Randy Goebel, and Ken Satoh. 2015. COLIEE-2015: evaluation of legal question answering. In *Ninth International Workshop on Juris-informatics*

Table 9: Technical category statistics of questions, and correct answers of submitted runs for each category in correct answer ratios. Alphabetical letters stand for team names; a: DBSE, b: EVORA1, c: EVORA2, d: EVORA3, e: IITP, f: JNLP: t=78, g: JNLP(t=85), h: JNLP(t=98), i: KIS_3module, j: KIS_dic, l: TRAttn, m: TRSimFeat, n: UA_Ex, o: UA_Go, respectively. Team columns stand for their number of correct answers for corresponding category.

category	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
Conditions	0.57	0.53	**0.41**	0.46	0.59	0.48	0.51	0.55	0.63	0.60	0.57	0.58	0.53	**0.69**	0.61
Person role	0.55	0.56	**0.36**	0.39	0.61	0.52	0.53	0.58	0.65	0.59	0.62	0.53	0.52	**0.70**	0.56
Person relationship	0.55	0.56	**0.36**	0.39	0.61	0.52	0.53	0.58	0.65	0.59	0.62	0.53	0.52	**0.70**	0.56
Negation	0.61	0.55	0.45	0.45	0.59	0.50	**0.43**	0.59	0.59	**0.66**	0.52	0.52	0.50	0.59	0.55
Entailment	0.55	0.42	0.55	0.45	0.48	0.45	**0.30**	0.48	**0.64**	0.58	0.58	0.55	0.48	0.61	0.55
Dependency	0.43	0.50	**0.36**	0.39	0.50	**0.36**	0.61	0.57	0.68	0.61	0.64	0.68	0.43	**0.75**	0.61
Ambiguity	0.42	0.54	**0.35**	0.54	0.46	0.38	0.58	0.65	0.58	0.58	0.42	0.46	0.54	**0.65**	0.50
Legal terms	0.50	0.46	0.46	0.58	0.50	**0.38**	0.58	**0.75**	0.54	0.50	0.63	0.63	0.54	0.71	0.63
Anaphora	0.55	0.59	**0.41**	0.55	0.59	0.59	0.59	0.59	0.73	0.55	0.64	**0.41**	**0.41**	**0.82**	0.64
Verb Paraphrase	0.52	0.57	**0.38**	**0.38**	**0.71**	0.48	0.48	0.52	0.48	0.62	0.52	0.57	0.48	0.62	0.43
Morpheme	0.72	0.39	**0.33**	0.44	0.72	0.72	0.61	0.50	0.67	0.61	0.72	**0.78**	0.67	**0.78**	**0.78**
Case role	0.47	0.59	**0.35**	0.47	**0.71**	0.41	0.47	0.59	0.53	0.41	0.41	0.65	0.47	0.53	0.47
Predicate argument	**0.36**	0.50	0.43	0.50	0.71	0.50	0.50	0.57	0.64	0.50	0.64	0.64	**0.36**	**0.79**	0.50
Article search	0.45	0.55	0.36	0.64	0.73	0.64	0.55	**0.27**	0.73	0.73	0.45	0.36	0.36	**0.91**	0.64
Paraphrase	0.36	0.55	0.45	0.64	0.27	**0.09**	0.64	0.64	0.82	0.27	**0.91**	0.36	0.64	0.73	0.45
Itemized	**0.75**	0.50	0.63	**0.38**	0.63	0.50	**0.38**	0.50	0.50	**0.75**	0.50	**0.75**	0.63	**0.75**	0.63
Normal terms	0.43	0.43	**0.29**	**0.71**	0.57	**0.29**	0.57	0.57	**0.71**	**0.71**	0.57	0.43	**0.29**	**0.71**	0.57
Calculation	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	0.50	1.00	1.00

(JURISIN 2015).

[11] Quoc Le and Tomas Mikolov. 2014. Distributed Representations of Sentences and Documents. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32 (ICML'14)*. JMLR.org, II–1188–II–1196. http://dl.acm.org/citation.cfm?id=3044805.3045025

[12] Craig Macdonald, Richard McCreadie, Rodrygo LT Santos, and Iadh Ounis. 2012. From puppy to maturity: Experiences in developing Terrier. In *Proceedings of the SIGIR 2012 Workshop in Open Source Information Retrieval*. 60–63.

[13] Kanika Madan, John Hudzina, Thomas Vacek, Frank Schilder, and Tonya Custis. 2019. Textual entailment using word embeddings and linguistic similarity. In *Proceedings of the 6th Competition on Legal Information Extraction/Entailment (COLIEE'2019).*

[14] Kim Mi-Young, Juliano Rabelo, and Randy Goebel. 2019. Statute Law Information Retrieval and Entailment. In *Proceedings of the 17th International Conference on Artificial Intelligence and Law (ICAIL'2019).*

[15] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

[16] Ha Thanh Nguyen, Vu Tran, and Le Minh Nguyen. 2019. A Deep Learning Approach for Statute Law Entailment Task in COLIEE-2019. In *Proceedings of the 6th Competition on Legal Information Extraction/Entailment (COLIEE'2019).*

[17] Guilherme Paulino-Passos and Francesca Toni. 2019. Retrieving Legal Cases with Vector Representations of Text. In *Proceedings of the 6th Competition on Legal Information Extraction/Entailment (COLIEE'2019).*

[18] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proc. of NAACL*.

[19] Juliano Rabelo, Kim Mi-Young, and Randy Goebel. 2019. Combining Similarity and Transformer Methods for Case Law Entailment. In *Proceedings of the 17th International Conference on Artificial Intelligence and Law (ICAIL'2019).*

[20] Kashyap Raiyani and Paulo Quaresma. 2019. Keyword Machine Learning Based Japanese Statute Law Retrieval and Entailment Task at COLIEE-2019. In *Proceedings of the 6th Competition on Legal Information Extraction/Entailment (COLIEE'2019).*

[21] Julien Rossi and Evangelos Kanoulas. 2019. Legal Information Retrieval with Generalized Language Models. In *Proceedings of the 6th Competition on Legal Information Extraction/Entailment (COLIEE'2019).*

[22] Trevor Strohman, Donald Metzler, Howard Turtle, and W. Bruce Croft. 2005. *Indri: a language-model based search engine for complex queries.* Technical Report. in Proceedings of the International Conference on Intelligent Analysis.

[23] Vu Tran, Minh Le Nguyen, and Ken Satoh. 2019. Building Legal Case Retrieval Systems with Lexical Matching and Summarization using A Pre-Trained Phrase Scoring Model. In *Proceedings of the 17th International Conference on Artificial Intelligence and Law (ICAIL'2019).*

[24] Sabine Wehnert, Sayed Anisul Hoque, Wolfram Fenske, and Gunter Saake. 2019. Threshold-Based Retrieval and Textual Entailment Detection on Legal Bar Exam Questions. In *Proceedings of the 6th Competition on Legal Information Extraction/Entailment (COLIEE'2019).*

[25] Masaharu Yoshioka, Yoshinobu Kano, Naoki Kiyota, and Ken Satoh. 2018. Overview of Japanese Statute Law Retrieval and Entailment Task at COLIEE-2018. In *The Proceedings of the 12th International Workshop on Juris-Informatics (JURISIN2018).* The Japanese Society of Artificial Intelligence,, 117–128.

[26] Masaharu Yoshioka and Zihao Song. 2019. HUKB at COLIEE 2019 Information Retrieval Task - Utilization of metadata for relevant case retrieval. In *Proceedings of the 6th Competition on Legal Information Extraction/Entailment (COLIEE'2019).*

COLIEE Case Law Competition Task 1: The Legal Case Retrieval Task

Rajaa EL HAMDANI
rajaa@justice.cool
Justice.cool
Paris, France

Aurore TROUSSEL
aurore@justice.cool
Justice.cool
Law and tax department, HEC
Paris, France

Claire HOUVENAGEL
claire@justice.cool
Justice.cool
Paris, France

ABSTRACT

With the availability of big volumes of digital legal documents, there is significant work carried out in the domains of Legal Information Extraction and Retrieval. Software systems of legal documents automatic processing have benefits for both legal experts and the large public. We describe our approach to the Legal Case Retrieval Task of the COLIEE challenge.

KEYWORDS

case law information retrieval, AI and law, Juris-informatics

ACM Reference Format:
Rajaa EL HAMDANI, Aurore TROUSSEL, and Claire HOUVENAGEL. 2019. COLIEE Case Law Competition Task 1: The Legal Case Retrieval Task.

1 INTRODUCTION

It is important for lawyers to access and search for case law in order to find relevant cases to their work. A manual search for supporting cases is a time-consuming process given the large amounts of available case law, hence the need for computer applications which accelerate this process. There has been important progress in information extraction (IE) and information retrieval (IR) techniques for general domain documents and also specific domain documents such as legal documents. There exist commercial systems that allow lawyers to retrieve similar legal cases to a query case. However, there is still big progress to be made to develop software systems that output lawyer-like answers. As mentioned in [3], a lawyer "rarely looks for, or even expects, clear answers. More often than not, he searches his database treatises, articles, statutes, cases, and other materials in order to construct legally acceptable arguments in the pursuit of one or more objective". Therefore a system that outputs lawyer like answers would output legal arguments extracted from relevant legal cases that the lawyer could use to support a new case at hand.

The Legal Case Retrieval task of the COLIEE competition could be considered as a component of the system mentioned above. The task aims at retrieving supporting cases for a base case from a corpus of case law. A supporting case is called noticed case in the rest of the paper. There have been many works to extract information from legal documents. Since the 1960s the main approach was Boolean searches using "and", "or", and "not" for matching strings [2]. Alternative approaches were later proposed to alleviate the limitations of string search, such as conceptual search and indexation of case law [9] known as conceptual IR. Rissland, et al. [16] proposed to combine both IR and case-based reasoning (CBR) in order to retrieve relevant legal cases. In recent work, they adapt general methods to specific legal domains. For example, [8] they extract information from legal documents related to commercial law using the layout information of documents. [5] extracts relevant information from Philippine Supreme Court decisions such as the crime, the plaintiff and the penalty in order to structure unstructured legal cases . They modify an existing generic information extraction system by using NLP techniques: named entity recognition, co-reference resolution, POS-tagging... Most of the works about legal information retrieval and information extraction develop systems that are going to be used by legal experts. The BEST project [17] is amongst the few works that study legal IR for the general public. They propose a system which allows searching in case law using laymen terms and also representing results in a comprehensible way.

In this work, we model the Legal Case Retrieval task as a binary classification problem. Since it is an imbalanced class classification problem, we used a variant of bagging classifier that re-samples each subset of the data before training each base estimator. We constructed a feature matrix by extracting relevant information from cases.

In the rest of the paper, we describe our approach and results on the test dataset.

2 EXPERIMENTS

In this section, we describe the pre-processing, the feature extraction steps and the machine learning algorithms that were applied to the legal case retrieval task.

2.1 Dataset

The challenge dataset is a corpus containing Federal Court of Canada case laws. The training dataset has 285 query case. A query case will be called a base case in the rest of the document. Each base case in the training dataset has its own pool of candidate cases and a list of noticed cases from the pool. The test dataset has 61 base case, such that each base case has its pool of 200 candidate cases without a mention of the noticed cases.

We carried a preliminary analysis of the structure of Canada case law in order to know how to extract information that could be relevant to the task of extracting noticed cases. Here we describe the main findings of this analysis [1]. At the top of each case we find the legal citation for the case:

COLIEE 2019, June 21, 2019, Montreal, Quebec

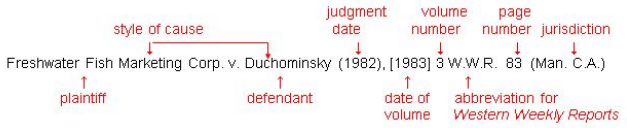


Figure 1: Example of legal citation [1]

The first part of the legal citation is the "style of cause" which is composed of the plaintiff's name and then the defendant's name. The theme of the case could be inferred from the defendant name if the defendant is a public department. For example, if the defendant is The Minister of Citizenship and Immigration we could expect that the case would be about immigration issues and that a case is more likely to cite past cases with the same defendant.
The judgment date and the date of the volume are useful since a case can only be cited by a future judgment.
The abbreviation W.W.R stands for the name of the volume in which the case was cited, and we assume that similar cases are published in the same volume.
A legal citation appears also in the text of the case, in which case it is called an in-text citation. In-text citations are interesting because they come after a sentence that summarizes the legal rule of the case and therefore indicates why a case was cited in another case. The name of the judge that decided the case is found just before the text of the decision in the following manner:

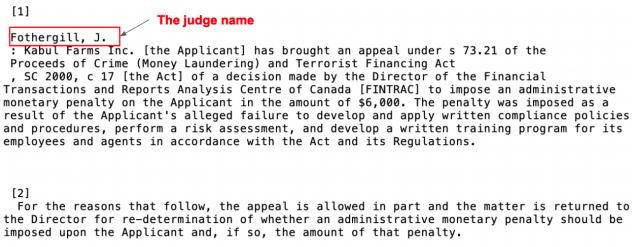


Figure 2: Example of a judge name as mentioned in a case

Another important section at the beginning of the case is the section containing legal topics. A legal topic is a specific legal subject meaningful to legal researchers. Furthermore, legal topics are arranged into a tree hierarchy. Legal topics are used by legal researchers to find similar cases, so it is likely that a noticed case share some legal topics with the base case.

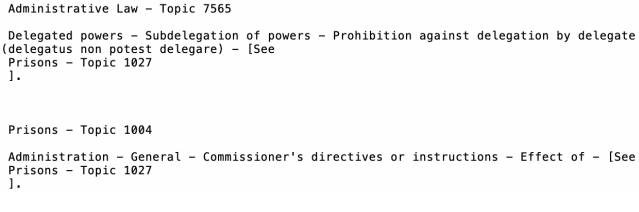


Figure 3: Examples of legal topics as mentioned in a case

Each topic has [12]:

- An id of 4 digits
- The title of the area of law to which it belongs, for example: Administrative Law or Prisons
- A list of keywords defining the legal topic

2.2 Dataset Pre-processing

The pre-processing step prepares the text of cases for the learning step. Cases are first segmented into phrases using regular expressions. Some cases contain the French translation of the case, so we had to remove french sentences using **langdetect** [1] tool. As mentioned before, the feature matrix is constructed by extracting information from both the base case and the candidate case. The results of the preliminary analysis, described in the section before, helped in defining the regular expressions to extract some metadata from cases. Here is the list of the features that were extracted for both type of cases:

- The jurisdiction of the case
- The date of the case
- The respondent of the case
- A boolean value indicating whether the respondent is a government department
- The reporter name
- The reporter volume
- The name of the case judge
- The IDs of legal topics
- The titles of legal topics
- The titles of statutes mentioned in the case
- The reference letters of the applicant
- A boolean value indicating if the case is unedited
- A boolean value indicating if the case is being edited

All these features are categorical features, except for the year and the Boolean values, so all categorical features were one hot-encoded.

Table 1: Number of unique values for each categorical feature

Topic	Judge	Jurisdiction	Respondent name	Reporter volume	Reporter name	Reference letters	Statutes titles
1111	155	3	416	449	21	8	267

To reduce the dimensionality of the feature matrix we kept only:

- Topics that occurred in at least 3 cases
- Respondents that appeared in at least 6 cases

Therefore the resulting feature matrix has 4860 columns and $57000 = 285 * 200$ training samples since we have 285 base case and 200 candidates per case.

2.3 Machine learning model

We model the legal case retrieval problem as a binary classification problem. Base cases are paired with each one of their candidate cases. If the candidate case of the pair is a noticed case for the base case then the pair is assigned a positive label, otherwise, it is assigned a negative label. This classification dataset suffers from imbalanced classes, such that the positive class is only 2.6% of the entire training dataset.

[1] https://github.com/Mimino666/langdetect

In this section, we describe the models we tested. Each model is evaluated using grouped K-fold cross-validation to ensure that the training dataset and the test dataset do not have common base cases since we expect the model to generalize well on unseen base cases. The hyperparameters search is carried out with a random search such that the best hyperparameters values are the ones with the best F1-score. The importance of recall vs precision depends on the context of the information retrieval task. Most users of web search engines want to have relevant documents on the first page of the search results, therefore high precision is important and low recall is tolerated. However, professionals such as paralegals are interested in getting almost all the relevant documents and they tolerate a reasonable number of returned irrelevant documents, therefore the recall is more important than the precision in that case. We chose to select hyperparameters and models according to the F1-score since the COLIEE challenge uses precision, recall and F1-score as evaluation metrics and does not give any instruction about the importance of each metric.

Most of the classification algorithms give poor performance when applied to this type of datasets because the formulation of the objective function does not give specific attention to the minimal class [4]. We first reduce the imbalance by removing training pairs where the candidate case is posterior to the base case since noticed cases are always anterior to the base case. With this method, the positive class ration increases to 3.3% but the dataset is still highly imbalanced.

There are two main techniques to handle imbalanced datasets:

- Cost sensitive learning
- Sampling techniques:
  - Under-sampling of the majority class
  - Over-sampling of the minority class
  - Combining under-sampling and over-sampling

Under-sampling proved its efficiency for imbalanced class classification problems [7], but this technique ignores an important number of the majority class examples. To deal with this limitation, we chose to test some ensemble learning algorithms that are built for imbalanced datasets and we compared them with an ensemble learning technique without under-sampling:

2.3.1 Balanced Bagging classifier. Bagging classifier is a bootstrap ensemble method. Each base classifier is trained on a random subset of the original dataset which is generated with or without replacement and has the same size as the original dataset. These randomly drawn subsets of data are not balanced and therefore the base estimator will have difficulty learning the minority class. Balanced bagging classifier is a modified version where each random subset is balanced by under-sampling the majority class so it has the same size as the minority class. We tested this approach using a random under-sampling strategy: first, a random subset is generated with replacement, then samples of the majority class are removed randomly until the two classes are equal. Three different base classifiers were tested: decision trees, random forest, and extremely randomized trees.

2.3.2 Balanced random forest. Random forest classifier has poor performance on imbalanced class classification problems. In [10] they propose two variants of random forest in order to improve its performance on imbalanced datasets.

- Weighted random forest
- Balanced random forest

Weighted random forest is a cost-sensitive learning algorithm where the miss-classification of minority class is penalized by introducing class weights. Class weights are introduced into the algorithm by two means. First, splits are found by using a weighted version of the Gini criterion. Second, the predictions are made by a weighted majority vote such that the vote for a giving class is multiplied by its weight.

Balanced random forest trains each tree on a balanced random subset. Each subset is created by randomly drawing a sample from the minority class and randomly drawing the same number from the majority class. For this work, we used the balanced random forest variant since it uses an under-sampling technique.

2.3.3 Easy Ensemble classifier. Easy Ensemble classifier was first introduced in [13]. Easy Ensemble classifier trains T separate base classifiers $H_i (1 \leq i \leq T)$, each on a randomly sampled subset. Each subset is generated by keeping all the samples from the minority class and randomly sampling from the majority class such that the size of the sampled majority class is equal to the size of the original size of the minority class. AdaBoost learns the T base classifiers with s_i weak classifiers, corresponding weights $\alpha_{i,j}$, and the ensemble's threshold θ_i. The formula of the learnt base classifier H_i is:

$$H_i(x) = sgn(\sum_{j=1}^{s_i} \alpha_{i,j} h_{i,j}(x) - \theta_i)$$

. Easy Ensemble treats each weak classifier $h_{i,j}$ as a feature extracted from the training subset. The final classifier is constructed by combining all these features:

$$H(x) = sgn(\sum_{i=1}^{T} \sum_{j=1}^{s_i} \alpha_{i,j} h_{i,j}(x) - \sum_{i=1}^{T} \theta_i)$$

By combining features $h_{i,j}$ extracted from different random subsets of the majority class, Easy Ensemble makes sure to incorporate information from different samples of the majority class.

2.3.4 Experimental details. As mentioned above, random search is used to tune the hyperparameters of each model. We used the following hyperparameters' distributions:

- Balanced Bagging classifier:
  - Base estimators: random forest, decision trees, and extremely randomized trees
  - Number of base estimators: a uniform discrete random variable in the interval $[10, 200]$
  - The maximum fraction of features to train each base estimator : a uniform continuous random variable in the interval $[0.3, 1]$
- Bagging classifier without an under-sampling strategy: the same distribution of hyperparameters as for the Balanced Bagging classifier.
- Balanced random forest classifier:
  - Number of estimators: a uniform discrete random variable in the interval $[10, 200]$

  - The maximum depth of the tree: a uniform discrete random variable in the interval $[1, 30]$
  - The minimum fraction of samples required to split an internal node: a uniform continuous random variable in the interval $[0.1, 1]$
  - The minimum fraction of samples required to be at a leaf node: a uniform continuous random variable in the interval $[0.1, 0.5]$
  - The number of features to consider when looking for the best split: the total number of features, or 1000, or the square root of the total number of features, or the binary logarithm of the total number of features
- Easy Ensemble classifier:
  - Number of base estimators: a uniform discrete random variable in the interval $[10, 200]$

The implementation of tested models comes either from scikit-learn [15] or imbalanced-learn package [11].

2.4 Results

The model that gave the best performance on the training dataset, using grouped K-fold cross-validation, is the Balanced Bagging classifier with extremely randomized trees as the base classifier. Table 2 shows the results of tested models using a grouped 5-fold cross-validation. The reported scores are the average of scores obtained for each fold.

Table 2: Results of the compared methods using cross-validation

Model	Precision	Recall	F1-score
Balanced Bagging-Decision trees	0.1867	0.7269	0.2960
Balanced Bagging-Random forest	0.2142	0.6933	0.3250
Balanced Bagging-Extremely randomized trees	0.3335	**0.9492**	**0.4925**
Balanced random forest	0.0668	0.7285	0.1221
Easy Ensemble	0.1320	0.7870	0.2248
Bagging-Random forest	0.9619	0.1737	0.2932
Bagging-Extremely randomized trees	**0.9714**	0.2237	0.3676

Balanced Bagging, with extremely randomized trees as a base classifier, has the best F1-score. Therefore, we selected it to compute predictions on the test dataset. The following table shows the results of the selected model on the submitted test runs:

Table 3: Results on the test dataset

Precision	Recall	F-score
0.2119	0.5848	0.3110

As we can see in table 2, all the models have a low performance on this task. A second observation is that all the models that used under-sampling have an approximately high recall at the expense of low precision, while models without under-sampling (Bagging-Random Forest and Bagging-Extremely Randomized Trees) have high precision and low recall. An explanation for this observation is that under-sampling has a warping effect over the posterior distribution for each of the class probabilities. This effect was studied in [6], where they demonstrated that the posterior probability of belonging to the minority classes increases when under-sampling increases according to the following formula:

$$p_s = \frac{p}{p + \beta(1-p)}$$

where p_s is the posterior probability of belonging to the minority class after under-sampling, p is the true probability of belonging to the probability class in the original dataset, and β is the probability of selecting a negative instance with under-sampling. You can see in figure 4 the relationship between p_s and p. As shown

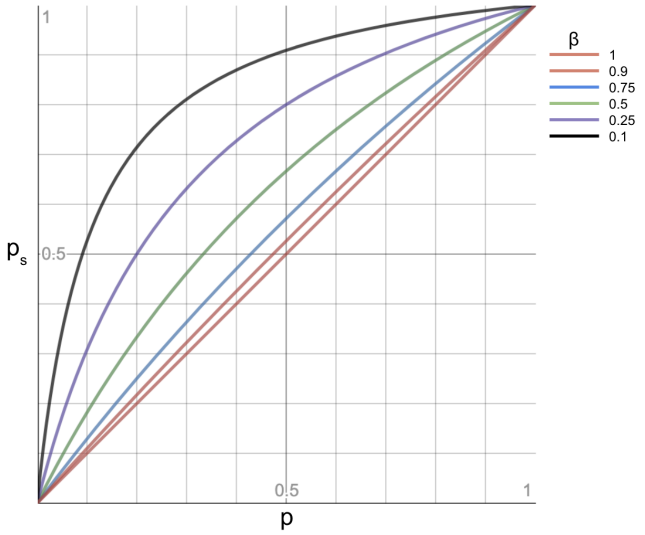


Figure 4: Relationship between p_s and p for different values of β. The under-sampling effect is inversely proportional to β.

in figure 4, a strong under-sampling of the dataset increases the posterior probability p_s of belonging to the minority class. This observation may explain the obtained results. In all the models that use under-sampling to balance the dataset, the under-sampling is strong because the dataset is highly imbalanced. Therefore, the warping effect is strong which is coherent with the high recall and low precision of these models.

2.4.1 Analysis of feature importance. In our approach we only used meta-data about cases as features, so we would like to know what features the model learned as important. The selected model can be seen as an ensemble of decision trees and therefore feature importance could be computed by averaging the feature importance of each tree [14]. Figure 3 is a plot of the importance of the 50 first important features. The first most important feature is the year of the judgment of the candidate case. Figure 6 compares the distribution of the difference between the base case year and the candidate case year for the two classes on the training dataset. We can see from the figure that for noticed candidate cases the years' difference are much smaller than for candidate cases which

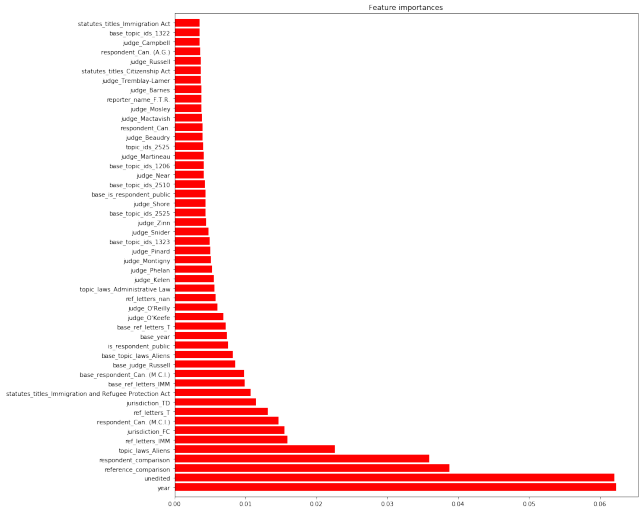


Figure 5: Feature importance of the 50 first important features

are unnoticed, such that for a difference larger than 17 years the candidate cases aren't noticed. This behavior could be explained by the fact that the law changes with time, and that the importance of cases decreases with time such that a case impacts future cases in a limited time frame.

3 CONCLUSION

We proposed an approach that models the Legal Case Retrieval Task as a binary text classification task. We used features that were only extracted from the header of the decision, and which do not include any description of the case facts. The classification dataset is highly imbalanced, so we used machine learning models that leverage both under-sampling techniques and ensemble techniques to deal with limitations of classical classification models when applied to extremely imbalanced datasets. Our approach shows that there is some correlation between meta-data features and whether a candidate case is a noticed case. For example, the release year of the candidate case is a strong indicator of whether it is a noticed case. An explanation is that the generated pool of candidate cases does not take into account the temporal relationship between the base case and its noticed cases. A suggestion is to only include in the pool candidate cases with a release year anterior to the base case within a likely time frame, in order to encourage retrieval models to rely more on feature related to the legal merit of the case. Under-sampling techniques have a warping effect on the posterior probability which results in a high recall. Therefore, under-sampling methods might be interesting if the recall has more importance than precision.

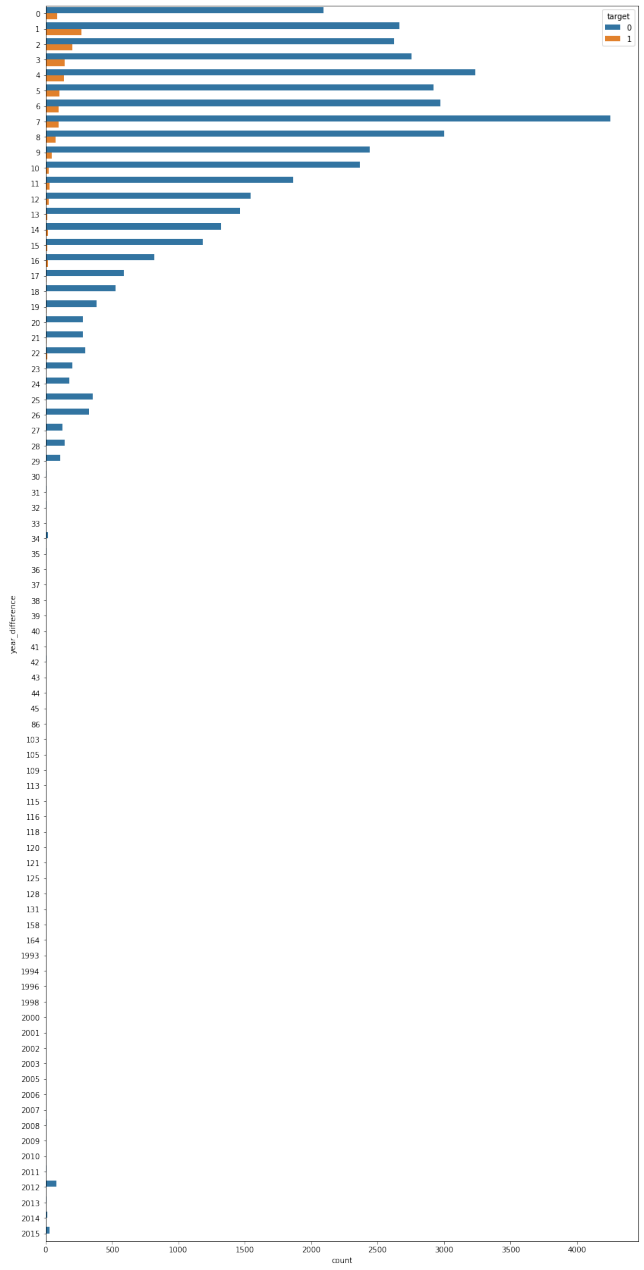


Figure 6: Distribution of the difference between the year of the base case and the year of the candidate case

REFERENCES

[1] Courthouse Libraries BC. 2015. How Do I Find Case Law? https://wiki.clicklaw.bc.ca/index.php?title=How_Do_I_Find_Case_Law?

[2] Trevor Bench-Capon, Michał Araszkiewicz, Kevin Ashley, Katie Atkinson, Floris Bex, Filipe Borges, Daniele Bourcier, Paul Bourgine, Jack G Conrad, Enrico Francesconi, et al. 2012. A history of AI and Law in 50 papers: 25 years of the international conference on AI and Law. *Artificial Intelligence and Law* 20, 3 (2012), 215–319.

[3] Bruce G Buchanan and Thomas E Headrick. 1970. Some speculation about artificial intelligence and legal reasoning. *Stan. L. Rev.* 23 (1970), 40.

[4] Nitesh V Chawla, Nathalie Japkowicz, and Aleksander Kotcz. 2004. Special issue on learning from imbalanced data sets. *ACM Sigkdd Explorations Newsletter* 6, 1 (2004), 1–6.

[5] Tin Tin Cheng, Jeffrey Leonard Cua, Mark Davies Tan, Kenneth Gerard Yao, and Rachel Edita Roxas. 2009. Information extraction from legal documents. In *2009 Eighth International Symposium on Natural Language Processing*. IEEE, 157–162.

[6] Andrea Dal Pozzolo, Olivier Caelen, and Gianluca Bontempi. 2015. When is undersampling effective in unbalanced classification tasks?. In *Joint European*

Conference on Machine Learning and Knowledge Discovery in Databases. Springer, 200–215.

[7] Chris Drummond, Robert C Holte, et al. 2003. C4. 5, class imbalance, and cost sensitivity: why under-sampling beats over-sampling. In *Workshop on learning from imbalanced datasets II*, Vol. 11. Citeseer, 1–8.

[8] Matías García-Constantino, Katie Atkinson, Danushka Bollegala, Karl Chapman, Frans Coenen, Claire Roberts, and Katy Robson. 2017. CLIEL: context-based information extraction from commercial law documents. In *Proceedings of the 16th edition of the International Conference on Articial Intelligence and Law*. ACM, 79–87.

[9] Carole D Hafner. 1987. Conceptual organization of case law knowledge bases. In *Proceedings of the 1st international conference on Artificial intelligence and law*. ACM, 35–42.

[10] Taghi M Khoshgoftaar, Moiz Golawala, and Jason Van Hulse. 2007. An empirical study of learning from imbalanced data using random forest. In *19th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2007)*, Vol. 2. IEEE, 310–317.

[11] Guillaume Lemaître, Fernando Nogueira, and Christos K. Aridas. 2017. Imbalanced-learn: A Python Toolbox to Tackle the Curse of Imbalanced Datasets in Machine Learning. *Journal of Machine Learning Research* 18, 17 (2017), 1–5. http://jmlr.org/papers/v18/16-365.html

[12] LexisNexis. 2019. Legal Topics Feature. https://www.lexisnexis.com/help/global/CA/en_CA/addtopicstips.asp

[13] Xu-Ying Liu, Jianxin Wu, and Zhi-Hua Zhou. 2009. Exploratory undersampling for class-imbalance learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 39, 2 (2009), 539–550.

[14] Gilles Louppe. 2014. Understanding random forests: From theory to practice. *arXiv preprint arXiv:1407.7502* (2014).

[15] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.

[16] Edwina L Rissland and Jody J Daniels. 1995. A hybrid cbr-ir approach to legal information retrieval. In *Proceedings of the 5th international conference on Artificial intelligence and law*. ACM, 52–61.

[17] Elisabeth M Uijttenbroek, Arno R Lodder, Michel CA Klein, Gwen R Wildeboer, Wouter Van Steenbergen, Rory LL Sie, Paul EM Huygen, and Frank Van Harmelen. 2008. Retrieval of case law to provide layman with information about liability: Preliminary results of the best-project. In *Computable models of the law*. Springer, 291–311.

Legal Information Retrieval with Generalized Language Models

ILPS Participation to COLIEE 2019

Julien Rossi
j.rossi@uva.nl
University of Amsterdam
Amsterdam, Netherlands

Evangelos Kanoulas
e.kanoulas@uva.nl
University of Amsterdam
Amsterdam, Netherlands

ABSTRACT

This paper describes a new method to identify text pairwise relevance, in the context of the Case Law retrieval task from COLIEE 2019. This method combines text summarizing and a generalized language model in order to assess pairwise relevance. With still lots of possibilities for improvement and optimization, it achieves a competitive performance in the setting of the Retrieval for the Noticed Cases.

CCS CONCEPTS

• **Information systems** → **Information retrieval**; **Retrieval models and ranking**; **Relevance assessment**; *Content analysis and feature selection.*

KEYWORDS

legal text, case retrieval, document representation, ranking, neural language models, BERT

ACM Reference Format:
Julien Rossi and Evangelos Kanoulas. 2019. Legal Information Retrieval with Generalized Language Models: ILPS Participation to COLIEE 2019. In *Proceedings of COLIEE 2019 workshop: Competition on Legal Information Extraction/Entailment (COLIEE 2019).* ACM, New York, NY, USA, 4 pages.

1 INTRODUCTION

We focus on the Task 1 of the COLIEE 2019 competition: the Legal Case Retrieval Task. In this task, a given court case is considered as a query to retrieve supporting cases (also named 'noticed cases') for the query case. A noticed case supports the decision taken in the query case, although the final decision itself is irrelevant in our retrieval. What matters is the proximity of the legal themes that are tackled by the query case and the noticed cases. Each query case is given a collection of potential supporting cases (also named 'candidate cases'), these collections are provided labeled for the training dataset, and unlabelled for the unknown test dataset.

We translate this retrieval task into a ranking problem, which we formulate as a pairwise relevance classification problem, given the binary labels "Noticed" or "Not noticed". We form a pairwise semantic latent representation of the query case and the candidate case, that we submit to a Multi-Layer Perceptron. The ranking of documents is based on the score for the positive class. We study as well how to use these scores to produce the optimum search result with regards to Precision, Recall and F1-Measure.

In very recent years, Generalized Language Models have provided State of the Art performances in many tasks related to Natural Language Processing, by providing general pre-trained systems on top of which practitioners could build classifiers for specific tasks. This approach is the opposite of ad-hoc specific systems designed for specific tasks.

The central problem is the projection of words and documents into a latent vector space in which usual vector operations, such as the cosine similarity, would mirror semantic proximity as observed by a human reader. Word2Vec[9] could generate a fixed representation per word, given a corpus, but it could not account for the different context any word might be used in. Contextual embeddings, such as CoVE[8] and ELMo[10] solved for that weakness, and introduced a collection of task-specific models. While ULMFit[5] was the first to introduce the idea of having two separate training phases: namely a pre-training phase, where a Language Model is being learnt, followed by a fine-tuning phase, where a system built on top of the feature generation is trained on a task. OpenAI GPT[11] continued that idea based on the concepts behind Transformers[14] for feature generation, and unsupervised pre-training[3] as a pre-training phase. BERT[4] added a bidirectional context for word features, while adhering to the two-step principle of pre-training and fine-tuning. Even more recently, OpenAI GPT 2[12] and MT-DNN[7] continued to improve on most of the NLP tasks.

In this paper, we formulate our pairwise relevance task as a downstream task for which a pre-trained BERT system is getting fine-tuned.

2 METHODOLOGY

2.1 Text Preprocessing

We consider the corpus as the complete collection of query cases, and unique candidate cases from the training dataset, with an estimated size of around 10000 unique documents, a total of 26 million tokens from a vocabulary of 370000 unique terms.

BERT uses WordPiece[15] tokens as inputs. Existing pre-trained BERT models accept inputs up to 512 WordPiece tokens. By design, WordPiece will subdivide each token into multiple tokens, we consider in our preparation that this will double the number of tokens observed under a standard Punkt[6] tokenizer as implemented in NLTK[2]. Our observation of the training dataset, in Figure 1, shows that we have to consider that all documents will be longer than 512 tokens, even by a factor of up to 2 0.

For this reason, we introduce a summarization of the texts in the corpus, as implemented in gensim[13] using TextRank[1]. We

COLIEE 2019, June 21, 2019, Montreal, Quebec

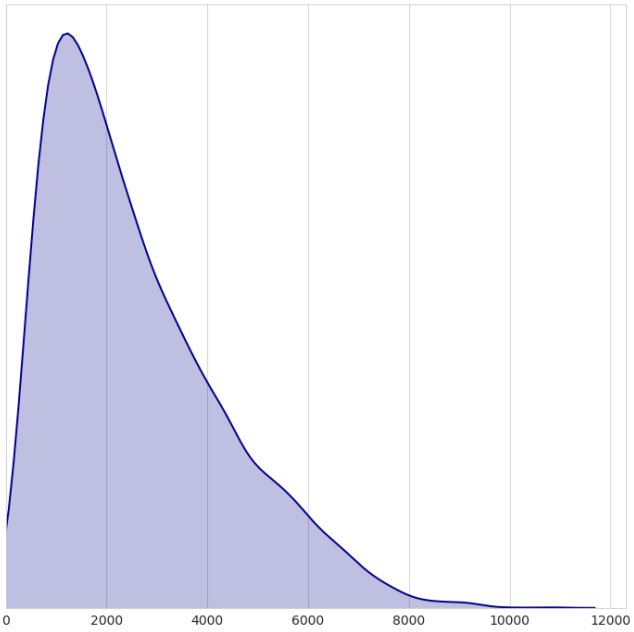


Figure 1: Distribution of the number of tokens per document.

choose to limit the size of the summary to 180 words, so a concatenated pair of texts will not be longer than 512 WordPiece tokens.

2.2 Documents Pairwise Embeddings

We instantiate a pre-trained BERT model (namely "Bert Base Uncased"), and use the hidden state of the 'CLS' token as the pairwise embedding for a pair of summarized query case and candidate case.

In this paper, we did not pre-train any further the BERT model with specific legal texts.

2.3 Relevance Assessment

We translate our pairwise relevance classification problem to the BERT fine-tuning for a downstream task of pairwise binary classification, by adding a Multi-Layer Perceptron that receives the pairwise embedding, and performs a binary classification.

We observe the distribution of the positive class over the dataset, we see that each query case has on average 4.8 noticed cases, and 90% of query cases have less than 10 noticed cases. The positive class is here the minority class, we balance the dataset by oversampling the positive class until a equal ratio is obtained for both classes.

We opted for a Mean Squared Error loss, and fine-tuned our model for 10 epochs. On our platform with one nVidia Tesla P40, with 24GB onboard RAM, the fine-tuning took 8 hours. For the binary classification task, we obtain an evaluation accuracy (on cases unseen during training) of 0.98, although we rely on the confusion matrix to assess the performance of the model, having in perspective that we will use the pairwise score for ranking purpose.

3 EXPERIMENTAL SETUP

The provided labeled dataset is split 75%/25% between training data and evaluation data. The dataset is split according to the cases, so that the cases in the evaluation dataset are not in the training dataset, so that it reflects properly the capacity of the system to generalize to unseen data. To be noted that, due to the setting of the task, both the case and the collection of candidates are unseen data[1].

After the fine-tuning step is finalized, the trained model computes the pairwise relevance score for each pair of query case and candidate case in the evaluation dataset, the score for the positive class is used to rank the candidate cases of each query case.

We proceed then to study the scores distribution, and evaluate optimum parameters for the selection of the candidate cases the system will return in response to the query case.

We observe the distribution of pairwise scores, with regards to the true class, in Figures 2 and 3. The distribution of the scores of the negative class is satisfactorily limited to the left, and we observe that the distribution of scores for the positive class has a more widespread mass. We expect this to have a negative impact on the recall of our system.

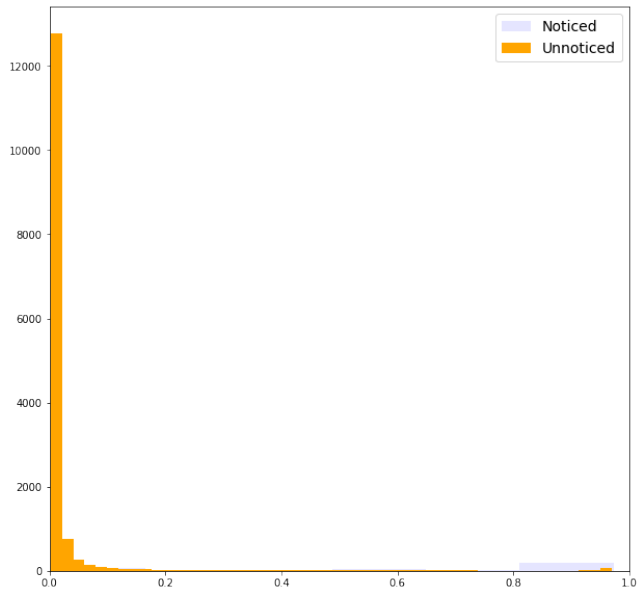


Figure 2: Distribution of Pairwise Relevance Score, for the Evaluation dataset. Negative class is highlighted

We consider two policies for building a list of returned candidate cases:

- Rank based: we will choose to return only the Top-N results, N being determined empirically from the observations made on the hold-out evaluation dataset
- Score based: we will choose to return all results with a score higher than a threshold value, determined empirically from the observations on the hold-out evaluation dataset

[1] although not all candidate cases are unique to only one query case, but we expect the statement to hold overall

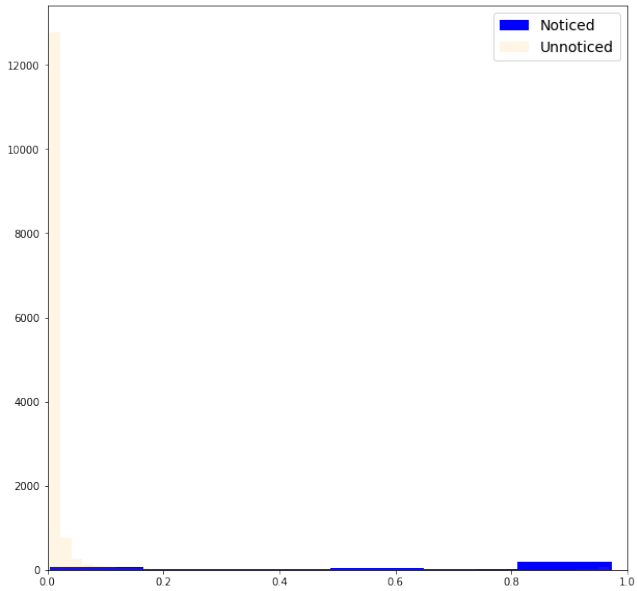


Figure 3: Distribution of Pairwise Relevance Score, for the Evaluation dataset. Positive class is highlighted

We opted for a Score-Based parameter for the BERT system, while our baseline is BM25 with a Rank-Based parameter. These parameters were used when predicting for the unseen and unlabeled test dataset. For BERT, we retained a value that was maximizing F1-score, and another value that was maximizing Precision.

4 RESULTS AND ANALYSIS

Our results are given in Table 1.

System		Evaluation Dataset			Test Dataset		
Name	Cut	P	R	F1	P	R	F1
BM25	6	0.47	0.55	0.51	0.47	0.52	0.49
BERT F1	0.946	0.72	0.47	0.57	0.68	0.43	0.53
BERT P	0.96	0.82	0.38	0.52	0.82	0.34	0.48

Table 1: Precision, Recall and F1-score.

The performance for the test set is the submitted result for the competition.

We observe that, for the selected metrics, the results on the unknown data (test dataset), based on parameters devised on a small portion of labeled data (evaluation dataset), are in line with the results on the evaluation dataset, which indicates a good capacity to generalize for our system.

As expected from the pairwise relevance score distribution, recall is negatively impacted by the widespread distribution of scores for the positive class, while precision suffers from the small proportion of pairs with a high score belonging to the negative class. The recall for both BERT systems is also lower than the BM25 baseline.

We consider the limitations of reducing a ranking problem to a pairwise relevance classification problem. The system learns to have a score on the right side of the decision threshold (0.5 in the binary classification setting), instead of learning that the score of any sample from the negative class should be lower than the score of any sample from the positive class. While this limitation is addressed by systems from the Learning to Rank family, this paper did not use the associated techniques.

We analyze further the submitted results, based on the disclosed golden labels. The unlabeled test dataset had, on average, 5.4 noticed cases per query case, which is slightly higher than 4.8 as observed for the labeled training dataset. We also plot the distribution of scores with regards to the True class in Figures 4 and 5, which we find almost identical to the distribution observed for the evaluation dataset. We conclude to the proper generalization of our system.

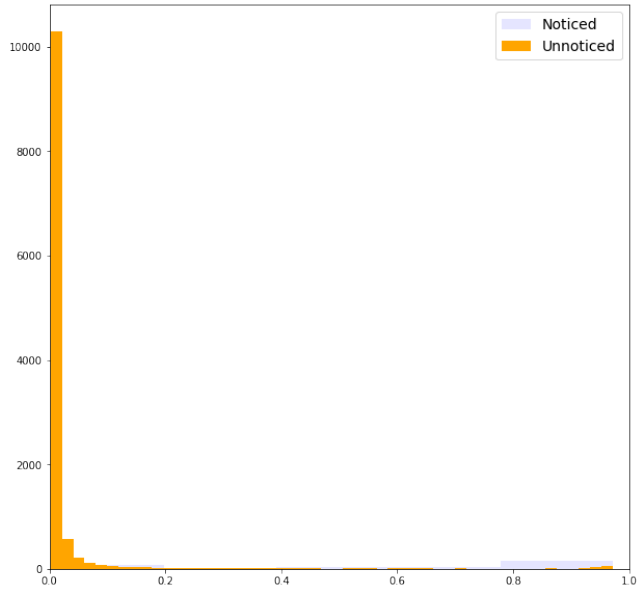


Figure 4: Distribution of Pairwise Relevance Score, for the Test dataset. Negative class is highlighted

5 CONCLUSION

We have demonstrated in this paper the ability of Generalized Language Models to perform properly, out of the box, in this particular setting of legal information retrieval. We did not establish a new State of the Art for this task, but we assume there are multiple opportunities for refinement and improvements.

We introduced the usage of summarization in order to keep the size of the input within the bounds imposed by BERT. We introduced the usage of BERT for the production of pairwise embeddings that are meaningful for a downstream task of relevance classification. Our policy for tuning the returned list of results have been introduced and has shown consistent results across two distinct datasets.

In our future work, we will certainly focus on refining and improving on the underlying BERT model, diversifying to other Generalized Language Models, as well as introducing advanced methods of ranking and training, such as Learning to Rank.

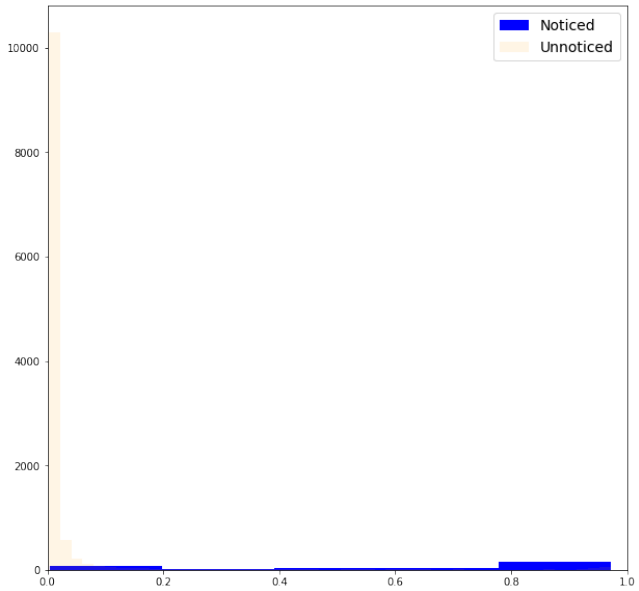


Figure 5: Distribution of Pairwise Relevance Score, for the Test dataset. Positive class is highlighted

REFERENCES

[1] Federico Barrios, Federico López, Luis Argerich, and Rosa Wachenchauzer. 2016. Variations of the Similarity Function of TextRank for Automated Summarization. *CoRR* abs/1602.03606 (2016). arXiv:1602.03606 http://arxiv.org/abs/1602.03606

[2] Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural Language Processing with Python*. O'Reilly Media.

[3] Andrew M. Dai and Quoc V. Le. 2015. Semi-supervised Sequence Learning. *CoRR* abs/1511.01432 (2015). arXiv:1511.01432 http://arxiv.org/abs/1511.01432

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *CoRR* abs/1810.04805 (2018). arXiv:1810.04805 http://arxiv.org/abs/1810.04805

[5] Jeremy Howard and Sebastian Ruder. 2018. Fine-tuned Language Models for Text Classification. *CoRR* abs/1801.06146 (2018). arXiv:1801.06146 http://arxiv.org/abs/1801.06146

[6] Tibor Kiss and Jan Strunk. 2006. Unsupervised Multilingual Sentence Boundary Detection. *Comput. Linguist.* 32, 4 (Dec. 2006), 485–525. https://doi.org/10.1162/coli.2006.32.4.485

[7] Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019. Multi-Task Deep Neural Networks for Natural Language Understanding. *arXiv preprint arXiv:1901.11504* (2019).

[8] Bryan McCann, James Bradbury, Caiming Xiong, and Richard Socher. 2017. Learned in Translation: Contextualized Word Vectors. *CoRR* abs/1708.00107 (2017). arXiv:1708.00107 http://arxiv.org/abs/1708.00107

[9] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. *CoRR* abs/1310.4546 (2013). arXiv:1310.4546 http://arxiv.org/abs/1310.4546

[10] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *CoRR* abs/1802.05365 (2018). arXiv:1802.05365 http://arxiv.org/abs/1802.05365

[11] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving Language Understanding by Generative Pre-Training. (2018). https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/language-unsupervised/language_understanding_paper.pdf

[12] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language Models are Unsupervised Multitask Learners. (2019). https://d4mucfpksywv.cloudfront.net/better-language-models/language_models_are_unsupervised_multitask_learners.pdf

[13] Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. ELRA, Valletta, Malta, 45–50. http://is.muni.cz/publication/884893/en.

[14] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. *CoRR* abs/1706.03762 (2017). arXiv:1706.03762 http://arxiv.org/abs/1706.03762

[15] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. *CoRR* abs/1609.08144 (2016). arXiv:1609.08144 http://arxiv.org/abs/1609.08144

Keyword & Machine Learning Based Japanese Statute Law Retrieval and Entailment Task at COLIEE-2019

Kashyap Raiyani and Paulo Quaresma
Computer Science Department
University of Évora, Portugal
(kshyp,pq)@uevora.pt

ABSTRACT

The worldwide Legal professionals are currently trying to get up-to-pace with the explosive growth in legal document availability through digital means. This drives a need for high-efficiency Legal Information Retrieval (IR) and Question Answering (QA) methods. The IR task, in particular, has a set of unique challenges that invite the use of semantic motivated NLP techniques. In this work, we propose a simple straight forward keyword-based method using algorithms like BM25, DLH13, LGD, Hiemstra_LM, TF_IDF and, DFRee for Legal Information Retrieval. The TF_IDF & LGD based model for Legal Information Retrieval was able to achieve the accuracy of 0.5334. For Recognizing Entailment, we proposed an end to end machine learning methods like Fasttext, CNN and, LSTM. The LSTM based model for Entailment Recognizing was able to achieve the accuracy of 0.6122.

CCS CONCEPTS

• **Information systems** → **Relevance assessment**; **Retrieval efficiency**.

KEYWORDS

Entailment, Legal Bar Exam, Retrieval and, Relevance

ACM Reference Format:
Kashyap Raiyani and Paulo Quaresma. 2019. Keyword & Machine Learning Based Japanese Statute Law Retrieval and Entailment Task at COLIEE-2019. In *Proceedings of COLIEE 2019 workshop: Competition on Legal Information Extraction/Entailment (COLIEE 2019).* ACM, Montréal, Canada, 6 pages.

1 INTRODUCTION

The ability to answer questions is a long-sought goal in the field of Natural Language Processing (NLP). Legal questions, in particular, pose a big challenge to current NLP techniques, due to their often complex syntactical structure and domain dependent terminology. As we experience explosive growth in legal document availability through digital means, the need for higher efficiency Legal Information Retrieval (LIR) and Question Answering (QA) methods becomes critical for the practice of the legal profession. The current increase in information analysis capabilities is not keeping up with such intense growth, leading to under-utilization of available legal resources and to the potential for information quality issues. This also brings up the matter of professional ethics and liability on law practice, due to the fundamental importance of relevant and correct information in legal practice.

Legal Information Retrieval (LIR) has always been a research topic within Artificial Intelligence & Law ('AI & Law'): in 'A History of AI & Law in 50 papers' [3] seven of those 50 papers have a relation to LIR. For the legal user though much research seems to be only remotely relevant for solving their daily problems in information seeking. The underrepresentation of legal practitioners within the AI & Law community might offer an explanation: "A lawyer has always the huge text body and his degree of mastery of a special topic in mind. For a computer scientist, a high-level formalization with many ways of using and reformulating it is the aim." [18]

Meanwhile, due to the advancements of the information era and the Open Data movement, the number of legal documents published online is growing exponentially, but accessibility and searchability have not kept pace with this growth rate. Poorly written or relatively unimportant court decisions are available at the click of the mouse, exposing the comforting myth that all results with the same juristic status are equal. An overload of information (particularly if of low-quality) carries the risk of undermining knowledge acquisition possibilities and even access to justice. Not surprisingly, LIR has been approached within AI & Law primarily with a focus on the conceptualization of legal information, while for daily legal work that might not always be the most effective approach.

A basic step into answering a legal question is retrieving the relevant legal information. For example, laws, facts and previous verdicts, and following their contents in order to decide their applicability to the given question. On a research point of view, it is a challenging problem considering laws are written with abstraction in mind, to cover most possible scenarios of any predicted situation. A semantically motivated NLP framework is then needed to allow abstraction and realization of the written law. With this understanding, approaches for both abstraction [2] and realization [4] have been proposed, with Machine Learning (ML) techniques taking an increasingly important role in language analysis [5, 8]. ML-based distributional semantics approaches have recently shown promising results in general domain IR [7, 16, 24]. However, they still perform poorly in the legal domain, due to the difficulty of training under datasets of relatively small size and a wide variety of topics with different vocabulary and semantics. For this reason, an approach combining lexical statistics and distributional semantics would be appropriate to leverage both accuracies in literal matching and the possibility of limited abstraction in the form of word/sentence embeddings. However, exploration of such combination has been limited, to the best knowledge of the authors.

COLIEE 2019, June 21, 2019, Montreal, Quebec

With the above understanding, we participated in COLIEE 2019 - Statute Law Competition. In particular, task 3 was regarding Legal Information Retrieval. The aim here is retrieving articles to decide on the appropriateness of the legal question. For this, the training and test data of the legal questions were collected from the Japanese bar exam by the organizing committee. All the questions and the Japanese civil law articles (1056 articles in total) were provided in two languages, Japanese and English. The English version of the law articles and questions was provided by the organizers. The aim of the task was to submit relevant articles for the questions using Japanese or English data. The maximum of three runs could be submitted.

On the other hand, task 4 was about Recognizing Entailment. Here, task 4 is to determine entailment relationships between a given problem sentence and an article sentence. The submitted answer should be yes or no regarding the given problem sentence. The same dataset as task 3 was provided with the 716 questions and labels for training.

In this work, we propose a simple straight forward key-word based method using algorithms like BM25, DLH13, LGD, Hiemstra_LM, TF_IDF and, DFRee for Legal Information Retrieval aimed at Question Answering. For Recognizing Entailment, we proposed an end to end machine learning methods like Fasttext, CNN and, LSTM. Here, for obtaining sentence representations, we used word embeddings.

The remainder of this paper is organized as follows: Section 2 presents related works and relevant results; Section 3 details the Legal Information Retrieval problem with system modeling and evaluation; Section 4 details the Recognizing Entailment problem with system modeling and evaluation; Section 5 presents the results and some discussion about the findings; Finally, Section 6 offers some concluding remarks.

2 RELATED WORK

Recent developments in Legal Information Retrieval (LIR) and Legal Question Answering (LQA) include the work of authors Liu, Chen and Ho [12], which presented a method called three-phase prediction (TPP) for retrieval of relevant statutes in Taiwanese criminal law, given queries presented in non-legal language. It employed a hierarchical ranking approach for law corpora, combining several Information Retrieval techniques, as well as Machine Learning and feature selection.

The use of distributional semantic representation into LQA has an interesting case in the work of Kim et. al. [11], in which an SVM-based ranking method using training features such as lemmatized words intersection, dependency pairs, and TF-IDF is used for LIR, while Recognition of Textual Entailment (RTE) was performed by a binary SVM classifier, trained on a set of features including semantic similarity calculated from word2vec [14] embeddings. This method won the combined LQA (LIR + RTE) COLIEE competition in 2016. The best-performing system for COLIEE 2018 used a tag cloud algorithm to select appropriate keywords to construct the query and the Terrier IR platform to retrieve the final results.

In the case of Entailment, all the previous method uses the sentences linguistically approach like calculating the similarity between predicates or defining the rules for condition, conclusion and, exception. Although, in COLIEE 2018, there was an attempt of using the combined deep neural network with additional features, and word2vec to retrieve the corresponding civil law articles which resulted as the last place with an accuracy of 0.4783.

Using the similar approach proposed in COLIEE 2018, the proposed system for task 3 is discussed in the next section.

3 STATUTE LAW RETRIEVAL TASK 3: DATASET, SYSTEM MODELING AND EVALUATION

This section is divided into three subsections namely, 1) Dataset and Processing, 2) System Modeling and, 3) Evaluation.

3.1 Dataset and Processing

For this task, questions related to Japanese civil law were selected from the Japanese bar exam. In a total of 716 questions as training and 98 questions as test data were provided. Apart from this, civil law with 1056 articles was provided. Using this civil law database, we created 4 corpora for indexing. Description for the same can be found as below:

(1) Corpus 1 - Original Civil Law Database
(2) Corpus 2 - Keyword Based Civil Law Database
(3) Corpus 3 - Summary Based Civil Law Database
(4) Corpus 4 - Extended Civil Law Database

For the retrieval purpose, using 716 questions, we created 4 different version of query-set. Description for the same can be found as below:

(1) Query 1 - Original 716 Questions
(2) Query 2 - Keyword Based 716 Questions
(3) Query 3 - Summary Based 716 questions
(4) Query 4 - Extended 716 questions

For batchevaluate, the qrels file was created which contained 914 entries.

3.2 System Modeling

For the system modeling, we used Terrier [15] platform and the algorithms like BM25, DLH13, LGD, Hiemstra_LM, TF_IDF and, DFRee with different weight. Terrier's indexing and retrieval architecture [15] could be found in the Figure 1 and 2 respectively.

The evaluation of different Corpus and Query discussed above (in Subsection 3.1) is shown next.

3.3 Evaluation

The Table 1, 2, 3 and, 4 show the accuracy over the training data. Among the table entries, EVORA1, EVORA2 and, EVORA3 represents the submitted runs. Mainly, the evaluations are done on the Corpus 1 - Original Civil Law Database and Corpus 2 - Keyword Based Civil Law Database with the Query 1 - Original 716 Questions and Query 2 - Keyword Based 716 Questions. Due to lack of time (late registration), we were not able to perform any experiments with Corpus 4 - Extended Civil Law Database and Query 4 - Extended 716 questions. In the case of the summary based Corpus and Query, the results were worst than Table 1. Here, the summarization and keyword extraction was done using LINGUAKIT [6] tool.

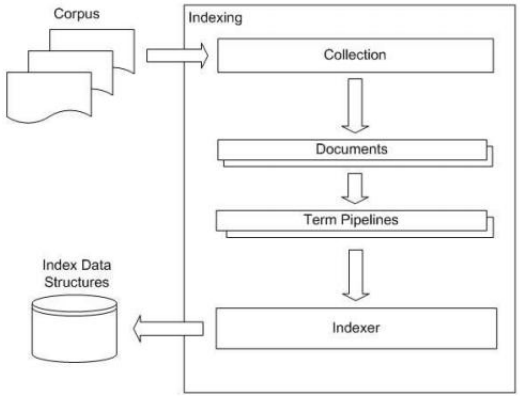


Figure 1: Indexing Architecture.

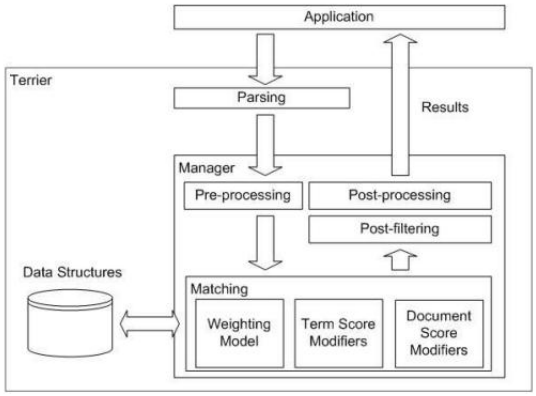


Figure 2: Retrieval Architecture.

Algorithm	Accuracy
BM25	0.5800
DLH13	0.5826
LGD	0.5815
Hiemstra_LM	0.5827
TF_IDF	0.5748
DFRee	0.5749

Table 1: Corpus 1 - Query 1.

Algorithm	Accuracy
BM25	0.5826
DLH13	0.5886
LGD	0.5865
Hiemstra_LM	0.5866
TF_IDF	0.5863
DFRee	0.5803

Table 2: Corpus 1 - Query 2.

Algorithm	Accuracy
BM25	0.5611
DLH13	0.5743
LGD	0.5727
Hiemstra_LM (EVORA3)	0.5739
TF_IDF	0.5626
DFRee	0.5658

Table 3: Corpus 2 - Query 1.

Algorithm	Accuracy
BM25	0.5905
DLH13	0.5887
LGD (EVORA2)	0.5990
Hiemstra_LM	0.5894
TF_IDF (EVORA1)	0.5887
DFRee	0.5918

Table 4: Corpus 2 - Query 2.

4 ENTAILMENT TASK 4: DATASET, SYSTEM MODELING AND EVALUATION

Even though less success of Deep Neural Network (DNN) in the Entailment task, we proposed DNN with Fasttext, CNN and, LSTM. Table 5 details the experimental properties. For the experiment, Tensorflow [1] was used.

Parameter	Value
MAX_SEQUENCE_LENGTH	400
MAX_NUM_WORDS	2000
EMBEDDING VECTOR	GLOVE [17]
EMBEDDING_DIM	300
VALIDATION_SPLIT	0.2

Table 5: Experimental Property.

Again, this section is divided into subsections namely, 1) Dataset and Processing, 2) Fasttext/CNN/LSTM Modeling and, 3) Evaluation.

4.1 Dataset and Processing

For this task, questions related to the civil law short answer (multiple choice) were derived from the Japanese legal bar exam. In a total of 716 questions as training and 98 questions as test data were provided. With this small training dataset (of only 716 question), an end to end deep neural network training was not possible. Therefore, to increase the size of the training dataset, we used the summation of below dataset entries.

(1) Dataset 1 - Questions - Answers
(2) Dataset 2 - Questions - Keyword Based Answers
(3) Dataset 3 - Keyword Based Questions - Answers
(4) Dataset 4 - Keyword Based Questions and Answers

Thus, resulting into 2864 entries of question and answers.

4.2 Fasttext Model

According to Joulin *et. al.* [10], a simple and efficient baseline for sentence classification is to represent sentences as a bag of words (BoW) and train a linear classifier, e.g., logistic regression or an SVM [20, 22]. However, linear classifiers do not share parameters among features and classes. For each example, y is a scalar that takes binary values ($\hat{y} \in [0, 1]$), while x is a vector of length N (where N is the number of features). Here, the output is a linear combination of the input features (i.e. a weighted sum of these features plus the biases).

$$\hat{y} = \left(\sum_i^N (w_i \cdot x_i) + b\right) \quad or \quad (\mathbf{w} \cdot \mathbf{x} + b)$$

where x and w are vectors of length N and the product $x \cdot w$ produces a scalar. As can be seen, a separate weight w_i for each input feature x_i is considered and these weights are independent by all means. This shows that there is no parameter sharing among features for linear classifiers. This possibly limits their generalization in the context of large output space where some classes have very few examples. Common solutions to this problem are to factorize the linear classifier into low-rank matrices [9, 13] or to use multilayer neural networks [19, 23]

Figure 3 shows a simple linear model with rank constraint architecture which was used for Entailment text classification (with the help of FastText).

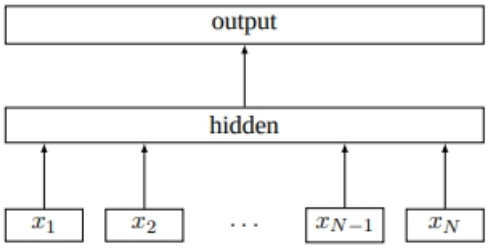


Figure 3: Model architecture of FastText for a sentence with N ngram features $x_1, \ldots, x_N$.

An Architecture similar to the cbow model of Mikolov *et. al.* [13], where the middle word is replaced by a label, here, the first weight matrix is taken as a look-up table over the words and then the word representations are averaged into a text representation, which is in turn fed to a linear classifier. The softmax function was used to compute the probability distribution over the predefined classes. For a set of N documents, this leads to minimizing the negative loglikelihood over the classes

$$-\frac{1}{N}\sum_{n=1}^{N} y_n log(f(BAx_n))$$

where x_n is the normalized bag of features of the n^{th} document, y_n the label, A and B the weight matrices. This model has trained asynchronously on multiple CPUs using stochastic gradient descent and a linearly decaying learning rate.

The parameters of training were sentences as an input(or N), learning rate of 1.0, hidden value of 50(or h), 25 epochs and hs loss function. With this, the system was able to learn 99.98% on a train set.

4.3 CNN Model

Convolutional Neural Network was initially developed for image processing community. Here, CNN achieved break-through results in recognizing an object from a pre-defined category (e.g., cat, bicycle, etc.). In the case of NLP tasks, i.e., when applied to text instead of images, we have a 1-dimensional array representing the text. Here the architecture of the ConvNets is changed to 1D convolutional-and-pooling operations.

1-D Convolutions over text - Given a sequence of words $w_{1:n} = w_1, ..., w_n$, where each is associated with an embedding vector of dimension d. A 1D convolution of width-k is the result of moving a sliding-window of size k over the sentence, and applying the same convolution filter or kernel to each window in the sequence, i.e., a dot-product between the concatenation of the embedding vectors in a given window and a weight vector u, which is then often followed by a non-linear activation function g. Figure 4 shows the architecture of the proposed model.

```
Training model.
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         (None, 400)               0
_________________________________________________________________
embedding_1 (Embedding)      (None, 400, 300)          600000
_________________________________________________________________
conv1d_1 (Conv1D)            (None, 393, 128)          307328
_________________________________________________________________
max_pooling1d_1 (MaxPooling1 (None, 78, 128)           0
_________________________________________________________________
conv1d_2 (Conv1D)            (None, 71, 64)            65600
_________________________________________________________________
max_pooling1d_2 (MaxPooling1 (None, 14, 64)            0
_________________________________________________________________
conv1d_3 (Conv1D)            (None, 7, 32)             16416
_________________________________________________________________
global_max_pooling1d_1 (Glob (None, 32)                0
_________________________________________________________________
dense_1 (Dense)              (None, 32)                1056
_________________________________________________________________
dense_2 (Dense)              (None, 2)                 66
=================================================================
Total params: 990,466
Trainable params: 390,466
Non-trainable params: 600,000
```

Figure 4: Proposed CNN Architecture.

4.4 LSTM Model

LSTM is used as Recurrent Neural Networks (RNN). The RNNs has a kind of internal dimension, that will be the dimension of h_i vectors. This dimension is constant over time. After one fed all inputs x_i into the RNN, the last output, h_t, is supposed to carry information about the whole sentence. This is theoretically true, but it shows weakness for long sequences. That's what LSTMs tries to solve. Here, we used a single layer of LSTM and Figure 5 shows the architecture of the proposed model.

4.5 Evaluation

All the three models are assessed over dataset mention in the Subsection 4.1. The Table 6, 7 and, 8 show the accuracy over the test gold

```
Training model.
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         (None, 400)               0
_________________________________________________________________
embedding_1 (Embedding)      (None, 400, 300)          600000
_________________________________________________________________
lstm_1 (LSTM)                (None, 64)                93440
_________________________________________________________________
dense_1 (Dense)              (None, 2)                 130
=================================================================
Total params: 693,570
Trainable params: 93,570
Non-trainable params: 600,000
```

Figure 5: Proposed LSTM Architecture.

data. Among the table entries, EVORA1, EVORA2 and, EVORA3 represents the submitted runs. In the evaluation tables, the "Poll" refers to the polling result over all the dataset runs (which were submitted official dry run).

Dataset	Correct	Accuracy
Dataset 1	57	**0.5816**
Dataset 2	51	0.5204
Dataset 3	54	0.5510
Dataset 4	50	0.5102
Poll (EVORA1)	50	0.5102

Table 6: Fasttext - Evaluation.

Dataset	Correct	Accuracy
Dataset 1	43	0.4388
Dataset 2	54	**0.5510**
Dataset 3	42	0.4286
Dataset 4	52	0.5306
Poll (EVORA3)	44	0.4490

Table 7: CNN - Evaluation.

Dataset	Correct	Accuracy
Dataset 1	42	0.4286
Dataset 2	43	0.4388
Dataset 3	45	0.4592
Dataset 4	60	**0.6122**
Poll (EVORA2)	47	0.4796

Table 8: LSTM - Evaluation.

5 RESULT AND DISCUSSION

For the Statute Law Retrieval, a total of 13 runs was submitted. Table 9 shows the overall performance of the proposed runs. For R5, R10 and, R30 proposed system outperformed all the submitted system.

Run	F2	MAP	R5	R10	R30	Rank
1	0.5334	0.6275	0.6694	0.7438	0.8512	3
2	0.5334	0.6170	0.6529	0.7438	0.8347	4
3	0.5289	0.6243	0.6529	0.7521	0.8347	5

Table 9: Statute Law Retrieval - Ranking.

From Table 9 it is observed that systems were able to achieve high MAP value.

For the Entailment Recognition, a total of 15 runs were submitted and Table 10 shows the overall performance of the proposed runs. Even though the submitted runs didn't perform well on the leaderboard, LSTM based approach was able to get an accuracy of 0.6122 (securing 3rd rank off-the-chart). This LSTM result was not submitted to the official dry run.

For task 4, we submitted the polling results over each dataset of all the mentioned (Deep Neural Network) methods (like Fasttext, CNN and LSTM). That is the reason our submitted runs were not able to perform better compared to individual run for each dataset.

Run	Correct	Accuracy	Rank
1	50	0.5102	12
2	44	0.4490	15
3	47	0.4796	14

Table 10: Entailment - Ranking.

6 CONCLUSION

Relevance, the basic notion of information retrieval "Is a thoroughly human notion and like all human notions, it is somewhat messy." As upheld in this paper [21], 'relevance' within legal information retrieval deserves specific attention and due to rapidly growing repositories. Because most LIR systems are designed by retrieval specialists without comprehensive domain knowledge, sometimes assisted by domain specialists with too little knowledge of retrieval technology, users are often disappointed by their relevance performance.

Two main conclusions can be highlighted. First, that keyword-based Information Retrieval still holds the upper hand compared to traditional/plain IR. Secondly, with the LSTM based experiments, we can strongly state that Entailment Recognition can be done using end to end deep neural network.

As a future scope to our work, for Information Retrieval, corpus expansion (i.e. solving internal reference) and query expansion can still deliver better results. On another side, with proper tuning of hyperparameter, end to end deep neural network can out-perform all the state of art methods.

ACKNOWLEDGMENTS

The authors would like to thank COMPETE 2020, PORTUGAL 2020 Programs, the European Union, and ALENTEJO 2020 for supporting this research as part of Agatha Project SI & IDT number 18022 (Intelligent analysis system of open of sources information for surveillance/crime control).

REFERENCES

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. https://www.tensorflow.org/ Software available from tensorflow.org.

[2] Kevin D Ashley and Edwina L Rissland. 2003. Law, learning and representation. *Artificial Intelligence* 150, 1-2 (2003), 17–58.

[3] Trevor Bench-Capon, Michał Araszkiewicz, Kevin Ashley, Katie Atkinson, Floris Bex, Filipe Borges, Daniele Bourcier, Paul Bourgine, Jack G Conrad, Enrico Francesconi, et al. 2012. A history of AI and Law in 50 papers: 25 years of the international conference on AI and Law. *Artificial Intelligence and Law* 20, 3 (2012), 215–319.

[4] Carlo Biagioli, Enrico Francesconi, Andrea Passerini, Simonetta Montemagni, and Claudia Soria. 2005. Automatic semantics extraction in law documents. In *Proceedings of the 10th international conference on Artificial intelligence and law*. ACM, 133–140.

[5] Michael Curtotti, Eric McCreath, Tom Bruce, Sara Frug, Wayne Weibel, and Nicolas Ceynowa. 2015. Machine learning for readability of legislative sentences. In *Proceedings of the 15th International Conference on Artificial Intelligence and Law*. ACM, 53–62.

[6] Pablo Gamallo and Marcos Garcia. 2017. LinguaKit: uma ferramenta multilingue para a análise linguística e a extração de informação. *Linguamática* 9, 1 (2017), 19–28.

[7] Debasis Ganguly, Dwaipayan Roy, Mandar Mitra, and Gareth JF Jones. 2015. Word embedding based generalized language model for information retrieval. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*. ACM, 795–798.

[8] Teresa Gonçalves and Paulo Quaresma. 2005. Is linguistic information relevant for the classification of legal texts?. In *Proceedings of the 10th international conference on Artificial intelligence and law*. ACM, 168–176.

[9] Schutze H. 1992. Dimensions of Meaning. In *Proceedings of the 1992 ACM/IEEE Conference on Supercomputing (Supercomputing '92)*. IEEE Computer Society Press, Los Alamitos, CA, USA, 787–796. http://dl.acm.org/citation.cfm?id=147877.148132

[10] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2016. Bag of Tricks for Efficient Text Classification. *CoRR* abs/1607.01759 (2016). arXiv:1607.01759 http://arxiv.org/abs/1607.01759

[11] mi-young Kim, Ying Xu, Yao Lu, and Randy Goebel. 2017. Question Answering of Bar Exams by Paraphrasing and Legal Text Analysis. 299–313. https://doi.org/10.1007/978-3-319-61572-1_20

[12] Yi-Hung Liu, Yen-Liang Chen, and Wu-Liang Ho. 2015. Predicting associated statutes for legal problems. *Information Processing and Management* 51, 1 (2015), 194–211. https://doi.org/10.1016/j.ipm.2014.07.003

[13] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. *CoRR* abs/1301.3781 (2013). arXiv:1301.3781 http://arxiv.org/abs/1301.3781

[14] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. *CoRR* abs/1310.4546 (2013). arXiv:1310.4546 http://arxiv.org/abs/1310.4546

[15] Iadh Ounis, Gianni Amati, Vassilis Plachouras, Ben He, Craig Macdonald, and Christina Lioma. 2006. Terrier: A high performance and scalable information retrieval platform. In *Proceedings of the OSIR Workshop*. 18–25.

[16] Hamid Palangi, Li Deng, Yelong Shen, Jianfeng Gao, Xiaodong He, Jianshu Chen, Xinying Song, and Rabab Ward. 2016. Deep sentence embedding using long short-term memory networks: Analysis and application to information retrieval. *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)* 24, 4 (2016), 694–707.

[17] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation. In *Empirical Methods in Natural Language Processing (EMNLP)*. 1532–1543. http://www.aclweb.org/anthology/D14-1162

[18] Edwina L Rissland and Jody J Daniels. 1995. A hybrid cbr-ir approach to legal information retrieval. In *Proceedings of the 5th international conference on Artificial intelligence and law*. ACM, 52–61.

[19] Collobert Ronan and Weston Jason. 2008. A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning. In *Proceedings of the 25th International Conference on Machine Learning (ICML '08)*. ACM, New York, NY, USA, 160–167. https://doi.org/10.1145/1390156.1390177

[20] Fan Rong-En, Chang Kai-Wei, Hsieh Cho-Jui, Wang Xiang-Rui, and Lin Chih-Jen. 2008. LIBLINEAR: A Library for Large Linear Classification. *J. Mach. Learn. Res.* 9 (June 2008), 1871–1874. http://dl.acm.org/citation.cfm?id=1390681.1442794

[21] Tefko Saracevic. 2007. Relevance: A Review of the Literature and a Framework for Thinking on the Notion in Information Science. Part II: Nature and Manifestations of Relevance. *J. Am. Soc. Inf. Sci. Technol.* 58, 13 (Nov. 2007), 1915–1933. https://doi.org/10.1002/asi.v58:13

[22] Joachims Thorsten. 1998. Text categorization with Support Vector Machines: Learning with many relevant features. In *Machine Learning: ECML-98*, Nédellec Claire and Rouveirol Céline (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 137–142.

[23] Xiang Zhang, Junbo Jake Zhao, and Yann LeCun. 2015. Character-level Convolutional Networks for Text Classification. *CoRR* abs/1509.01626 (2015). arXiv:1509.01626 http://arxiv.org/abs/1509.01626

[24] Guido Zuccon, Bevan Koopman, Peter Bruza, and Leif Azzopardi. 2015. Integrating and Evaluating Neural Word Embeddings in Information Retrieval. In *Proceedings of the 20th Australasian Document Computing Symposium (ADCS '15)*. ACM, New York, NY, USA, Article 12, 8 pages. https://doi.org/10.1145/2838931.2838936

A performance study on fine-tuned large language models in the Legal Case Entailment task

Hiroaki Yamada
yamada.h.ax@m.titech.ac.jp
School of Computing, Tokyo Institute of Technology
Meguro, Tokyo

Takenobu Tokunaga
take@c.titech.ac.jp
School of Computing, Tokyo Institute of Technology
Meguro, Tokyo

ABSTRACT

Deep learning based approaches achieved significant advances in various Natural Language Processing (NLP) tasks. However, such approaches have not yet been evaluated in the legal domain compared to other domains such as news articles and colloquial texts. Since creating annotated data in the legal domain is expensive, applying deep learning models to the domain has been challenging. A fine-tuning approach can alleviate the situation; it allows a model trained with a large out-domain data set to be retrained on a smaller in-domain data set. A fine-tunable language model "BERT (Bidirectional Encoder Representation from Transformers)" [5] was proposed and achieved state-of-the-art in various NLP tasks. In this paper, we explored the fine-tuning based approach in the legal textual entailment task using the COLIEE task 2 data set. The experimental results show that the fine-tuning improves the task performance, achieving $F1 = 0.50$ with COLIEE task 2 dry run data (Our group ID: TTCL).

CCS CONCEPTS

• **Information systems** → **Decision support systems**; *Data mining*; *Information retrieval*; *Specialized information retrieval*; • **Applied computing** → **Law**.

KEYWORDS

textual entailment, legal information processing, natural language processing, neural networks, deep learning, text classification, case law, AI and law

ACM Reference Format:
Hiroaki Yamada and Takenobu Tokunaga. 2019. A performance study on fine-tuned large language models in the Legal Case Entailment task. In *Proceedings of COLIEE 2019 workshop: Competition on Legal Information Extraction/Entailment (COLIEE 2019).* ACM, New York, NY, USA, 7 pages.

1 INTRODUCTION

COLIEE (Competition on Legal Information Extraction/Entailment) has been the most ambitious competitions in the area of statue law documents for years, and now it has the competition on the case law legal entailment task. The textual entailment in the legal domain is expected to be considerably more complex and challenging than other domains such as news texts and documents that are written in daily languages. The first case law legal textual entailment task in COLIEE'18 achieved at 0.26 in F1 measure [7]. The report of the best performing team in COLIEE'18 [14] concluded that the legal textual entailment task requires both deeper understanding of the problem and better embedding strategy of input sentences. The first point requires introducing domain-specific knowledge into models. The better understanding of the task domain enables to implement the better systems that utilized domain-specific characteristics such as, careful feature engineering, a citation network between cases, injecting external knowledge and building dedicated dictionaries. However, we leave this point for future research and devote this paper to the latter point – exploring the better way of embedding and encoding sentences in the legal domain.

The strategy of embedding and encoding sentences is a crucial part in neural network-based NLP. Word embedding like CBOW [10] and GloVe [13] has been a standard way to construct a vector representation of target input text for various classifiers in various NLP tasks. However, simple embedding is not sufficient for the legal entailment task to handle longer context and deeper semantics. We have to find a better way to encode entire sentences or paragraphs and to implement a classifier using the encoded features. One possible solution is a deep neural network-based model that can handle information beyond the word level, i.e., context. A bottleneck of this approach is that the model requires a large number of training instances. We can design a large model with many parameters that can handle deeper and longer context but it requires many instances to train the model properly, and it is hard to prepare large training data. The legal textual entailment is such a specialized task that annotator should have a certain level of legal knowledge to create stable and reliable training data. It makes building training data expensive. There are several solutions to the problem. Fine-tuning is one of the promising approaches. The fine-tuning retrains a model which is already trained with an out-domain data set with an in-domain data set. In COLIEE task 2, we can train a model with some large corpora (e.g., Wikipedia or Book corpus) in the first stage and later retrain it with the COLIEE task 2 data set. We can make the large model adapt to the smaller target data set by fine-tuning.

There have been several language models that are designed for a fine-tuning approach such as OpenAI GPT [15] and BERT [5] achieving state-of-the-art records in various kinds of NLP tasks including textual entailment. Our research question is to explore the effectiveness of fine-tuning in the legal textual entailment task and to identify its limitation.

COLIEE 2019, June 21, 2019, Montreal, Quebec

Table 1: Basic statistics on training data

	sentences	words
Avg. length of entailing paragraphs	3.8	104.1
Avg. length of entailed fragments	1.4	41.0

2 TASK AND DATA DESCRIPTION

We tackle COLIEE task 2, which is a legal textual entailment task in the Canadian case law system. Given a decision document Q (Base case) with an entailed fragment F and a relevant case document R (Noticed case) that includes n paragraphs as candidates ($P_R(n)$), a machine should identify paragraphs which entails Q (specifically F) from P_R. The number of entailing paragraphs is variable. We explain the task with an example in Figure 1 (Markers "ENTAILED FRAGMENT" and "ENTAILING PARAGRAPH" are added by author). The entailed fragment is Paragraph **[28]** of Base case 9, and the entailing paragraph was Paragraph **[11]** of Noticed case. This example has only one entailing paragraph.

The data source for the task is a collection of predominately Federal Court of Canada case law. In the provided corpus, there are the base case documents (Q), the entailed fragment (F), the paragraphs of noticed case documents ($P_R(n)$), and the paragraph IDs of the entailing paragraphs (answers) for the training data set. There is no entry for entailing paragraph IDs in the testing data set. Some parts of the base case documents are edited by the COLIEE organizer to replace some phrases with "FRAGNEBT_SUPPRESSED" to remove the obvious clues to resolve entailment. [11]

Table 1 shows the statistics of the data set. The average length of the entailed fragment was 104.1 in words and the average length of the entailing paragraph was 41.0 in words. Those numbers are calculated with the NLTK library [8]. The average number of the entailing paragraphs is 1.1 per case. Each entailing paragraph contains multiple sentences. The entailed fragments also can include multiple sentences though there is only one sentence in the example. As a model has to consider a broader context in this task, the problem is more difficult than conventional entailment tasks between sentences.

3 IMPLEMENTATION

We implemented baseline models with Support Vector Machine [4] and BERT-based models for COLIEE task 2. The provided base case documents include several editorial markers denoted by brackets, such as "[translation]", "[Emphasis added]", "[citations omitted]", "[my emphasis]", and "[End of document]". We removed those markers in the prepossessing stage.

3.1 Baseline models

Our baseline model is an SVM based implementation. The model takes the vector representations of both entailed fragment and target paragraph with auxiliary feature vectors and outputs a binary decision whether the pair has an entailing relationship. There are many ways of encoding sentences and paragraphs, and we prepared three ways of encoding with pre-trained embeddings and encoders — Skip-gram word2vec (W2V) [10], Neural Probabilistic Language

Base case (train-9)

...

This is an application for judicial review filed under subsection 72(1) of the Immigration and Refugee Protection Act, S.C. 2001, c. 27 (Act), against a decision by the Immigration Program Manager, Brian Ralph Hudson (Manager), of the Canadian Embassy in Beijing (embassy), People's Republic of China, dated November 18, 2004, denying the visa application of Yu Mei Zhang (applicant).

...

[27] In Mirzaii v. Canada (Minister of Citizenship and Immigration) , [2003] F.T.R. Uned. 100; 2003 FCT 213, Heneghan, J., states at paragraph 8 that a decision to issue a visa is a discretionary administrative decision and that it is therefore completely normal to rely on information gathered by an assistant. "In deciding whether to issue a visa, the Visa Officer is making an administrative decision involving the exercise of discretion. He was entitled to rely on information gathered by an assistant see Silion v. Canada (Minister of Citizenship and Immigration (1999), 173 F.T.R. 302. There is no evidence that Ms. Taheri did anything more than obtain information from the Applicant. The actual decision was made by the Visa Officer who was justified in relying on the facts obtained in the interview and recorded by Ms. Taheri in the CAIPS notes."

[28] in matters of administrative decisions, the rule of "he who hears must decide" does not apply. (ENTAILED FRAGMENT)

[29] I believe that the case law is clear on the fact that the manager need not personally conduct the interviews and the research. In conclusion, I do not think that in this case there was a violation of the principles of natural justice or of procedural fairness.

[30] THE COURT ORDERS THAT:
1. The application for judicial review be dismissed;
2. No question was submitted for certification.
Application dismissed.

Paragraphs from noticed case (14 paragraphs)

...

[10] This case concerns a decision to refuse a visa. The decision was clearly made by the visa officer, as he avers in his affidavit. This is supported by the affidavit of the IPO and further by the applicant's own affidavit which acknowledges that when she was advised of the decision to refuse her application she was told that the immigration officer made the decision, not the IPO.

[11] The decision is essentially an administrative one, made in the exercise of discretion by the visa officer. There is no requirement in the circumstances of this or any other case that he personally interview a visa applicant. There may be circumstances where failure to do so could constitute unfairness, but I am not persuaded that is the case here. Here the IPO did interview the applicant and reported on the results of that interview. That report was considered by the visa officer who made the decision. Staff processing and reporting on applications is a normal part of many administrative processes and it is not surprising it was here that followed. This is not a circumstance of a judicial or quasi-judicial decision by the visa officer which would attract the principle that he who hears must decide, or the reverse that he who decides must hear the applicant. (ENTAILING PARAGRAPH)

...

Figure 1: COLIEE Task2 query-answer sample

Model (NLM) [1], Universal Sentence Encoder(USE) [2]. Those pre-trained models are available online via Tensorflow hub[1].

W2V is a standard embedding method widely used in NLP tasks. The model is trained on the English Wikipedia corpus [9]. We tested two different models that have different vector sizes (250 and 500 dimensions). NLM is another token-based embedding technology which is built on a feedforward neural network language model. We used a model pre-trained on the English Google News corpus. We created a sentence embedding vector by normalizing weighted sum of the word embedding vectors. A sentence is represented by a 128-dimensional vector.

USE is designed to encode a sequence of more than one word. USE encodes a text into vector representations that can be used for various tasks including the text classification. We use the model to get the paragraph vectors of both entailed fragment and target paragraph. The pre-trained encoder models are available in two implementations from Tensorflow Hub. One is trained with a transformer encoder [16] and the other is trained with a deep averaging network [6] encoder. Both output a 512-dimensional vector.

As auxiliary features, we used the cosine similarity of the entailed fragment vector and the target paragraph vector, the relative position of the entailed fragment and the relative position of the target paragraph. Both relative positions are calculated as the position of the paragraph in the base case/noticed case documents divided by the total number of paragraphs in the documents.

We used the libsvm [3] based implementation provided by the scikit-learn machine learning library [12] for SVM.

3.2 BERT models

Our research question is whether language models trained with large corpora in general domains can be adapted to the legal text entailment task by the fine-tuning technique. We employ BERT in the present work. Several pre-trained models of BERT are available online[2] and they are trained on Wikipedia corpus and the BookCorpus [17]. Four pre-trained models are available with two kinds of options: two sizes (BERT-Base and BERT-Large) and two character casing options (Uncased and Cased). The Base models have 12 transformer blocks, hidden layers size of 768, 12 self-attention heads and 110M parameters while the Large models have 24 transformer blocks, hidden layers size of 1024, 16 self-attention heads, 340M parameters. The casing option means whether the text has been lowercased (uncased) or not. We use the uncased BERT-Base model in our fine-tuning experiments. The model which we used is a multi-layer Transformer encoder based architecture with approximately 110M parameters, which is similar to other Transformer based models like OpenAI GPT but employs the bidirectional self-attention.

BERT is capable of handling different language tasks such as sentence pair classification tasks, single sentence classification tasks, question answering tasks, and single sentence tagging tasks [5]. The COLIEE task 2 can be similar to a sentence pair classification task if we regard the entailed fragment and target paragraphs as "sentences". However, such straight-forward formalization is not appropriate since the pre-trained BERT model was trained for sequences up to 512 words. It means there are cases that we have to squeeze the fragments and paragraphs to that size, i.e. 512 words. According to our analysis, there are 54 cases that exceed 512 words. Instead of cutting the fragments and paragraphs, we break them into sentences and formalize the task as a simple sentence pairs task. In this new formalization, we firstly split fragments and paragraphs into sentences using NLTK [8] Punkt Sentence Tokenizer if there is more than one sentence, and make sentence pairs with every possible combination between sentences from entailed fragments and sentences from target paragraphs. Figure 2 visualizes the way of pairing. The BERT-based task 2 classifier takes each sentence pair and output whether the pair is in the relation of entailing. (Figure 3)

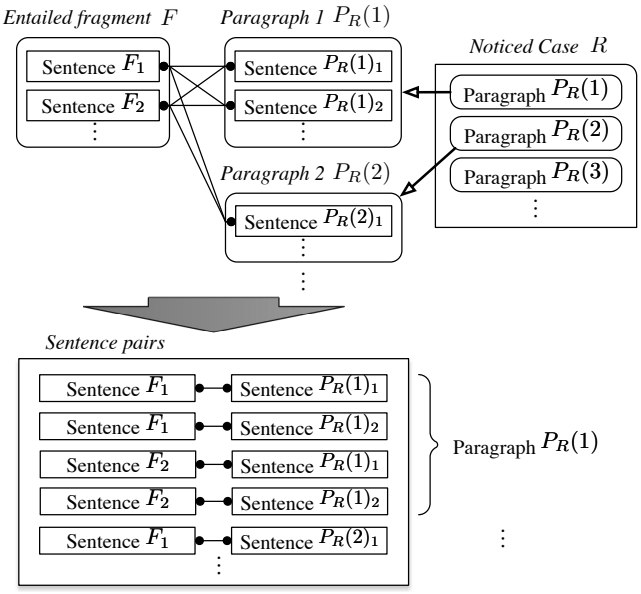


Figure 2: Visualized sentence pair preparation

To estimate an entailing relation between paragraph pairs from those of sentence pairs, we conduct the following post-processing. Given a paragraph pair, we regard the pair as an entailing pair if one or more sentence pairs across the paragraph pair are classified as "entailing."

4 EXPERIMENTS

4.1 Dry run experiments and results

Table 2: Grid search space for SVM hyperparameters

Hyper params	search space
C	1, 10, 100, 1,000
Kernels	Linear, RBF, Polynomial, Sigmoid
Gamma*	0.001, 0.0001
Degree*	2, 3, 4
Encoders	NLM, wiki500, wiki250, USE transformer, USE DAN

∗ If applicable

[1] https://tfhub.dev/
[2] https://github.com/google-research/bert

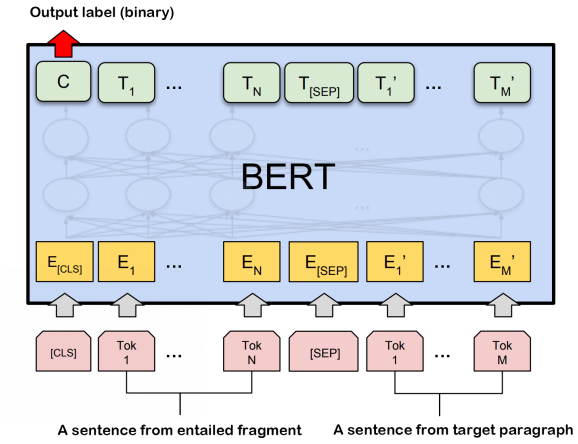


Figure 3: The model flow in COLIEE task 2 (the figure is based on and modified from [5])

We leave 20 base case documents out from the provided "train" data by the competition committee as a development set to tune hyperparameters and to select the optimal way of encoding. The remaining data is used as a training data set. Hyperparameter tuning is conducted using the grid search algorithm. Table 2 shows the search space. "NLM" stands for Neural Probabilistic Language Model based encoder. "wiki500" and "wiki250" are encoders that are based on W2V with different output vector sizes. The encoders output 500 and 250-dimensional vectors respectively. USE, universal sentence encoder has two implementations. "USE transformer" is a transformer based encoder and "USE DAN" is a deep averaging network based encoder.

As a result of tuning, we train SVM with the following parameters, $C = 10$, a linear kernel, and the USE DAN encoder.

We conducted experiments on the training data set with the five-fold cross validation to evaluate the performance of models. The data set includes 5,049 target paragraphs in total. We used an SVM model with tuned parameters and three BERT-based models with three different hyperparameter settings. Each BERT-based model allows different input sentence pair length (BERT-128, BERT-256 and BERT-512 take respectively 128, 256, 512 words as input). Also, we implemented an ensemble model BERT-vote using a simple voting method that the model will decide the output based on a majority prediction for each instance according to the three BERT-based models. Table 3 shows the results of the experiment. As for the BERT-based models, we report the results on sentence pairs based evaluation in addition to the standard paragraph based evaluation (results before merging) in Table 4. All BERT models show a significantly better result than the SVM-baseline model. BERT-vote was the best performing model among all model we implemented, achieving $F1 = 0.50$ while SVM-baseline was at $F1 = 0.22$.

4.2 Post-hoc analysis on dry run

Table 5 is a confusion matrix of the result from BERT-vote. We see that the model tends to produce more false positives than false negatives. Figure 4 shows a visualized comparison of predictions among the models in the experiment. Each square box represents a result of a prediction on each target paragraph. Blue square means true negative and green square means true positive while yellow square means false positive and red square means false negative. The figure includes only 51 cases of the provided "train" data and does not show the true negatives by all five models.

In the overall training data set (5,049 instances), 568 instances are correctly predicted by BERT-vote but not by SVM-baseline. On the other hand, 123 instances are correctly classified by SVM-baseline but not by BERT-vote. Neither SVM-baseline nor BERT-vote could correctly predict 109 instances, and we consider those instances were hard cases.

Figure 5 and Figure 6 are hard case examples from Base case 21. The true entailing paragraph is Paragraph 21, but all models failed to detect Paragraph **[21]** as entailing. Instead, BERT-256, BERT-vote, and BERT-512 predict Paragraph **[18]** and **[19]** as entailing. Interestingly, Paragraph **[18]** shares the same sentence, which is emphasized in boldface, with the entailed fragment of the Base case. The sentence should have been a misleading clue for the model in recognizing the entailment.

Table 3: Results on the training data set

Models	Precision	Recall	F1
SVM-baseline	0.14	0.52	0.22
BERT-128	0.37	0.54	0.44
BERT-256	0.36	0.66	0.46
BERT-512	0.36	0.59	0.45
BERT-vote	0.41	0.65	**0.50**

Table 4: Results on the training data set (sentence pairs)

Models	Precision	Recall	F1
BERT-128	0.43	0.12	0.18
BERT-256	0.41	0.14	**0.21**
BERT-512	0.42	0.12	0.19

Table 5: Confusion matrix for BERT-vote

		Predicted labels: Positive	Predicted labels: Negative	Total
True labels	Positive	118	64	182
	Negative	168	4,699	4,867
	Total	286	4,763	5,049

4.3 Formal run results

According to the results of the experiment on the training data set, we submitted three models, BERT-256, BERT-512 and BERT-vote to the formal run of COLIEE task 2. Those models are trained with all instances of the provided "train" data. Table 6 shows the results of the formal run. The best model was BERT-256 achieving $F = 0.53$.

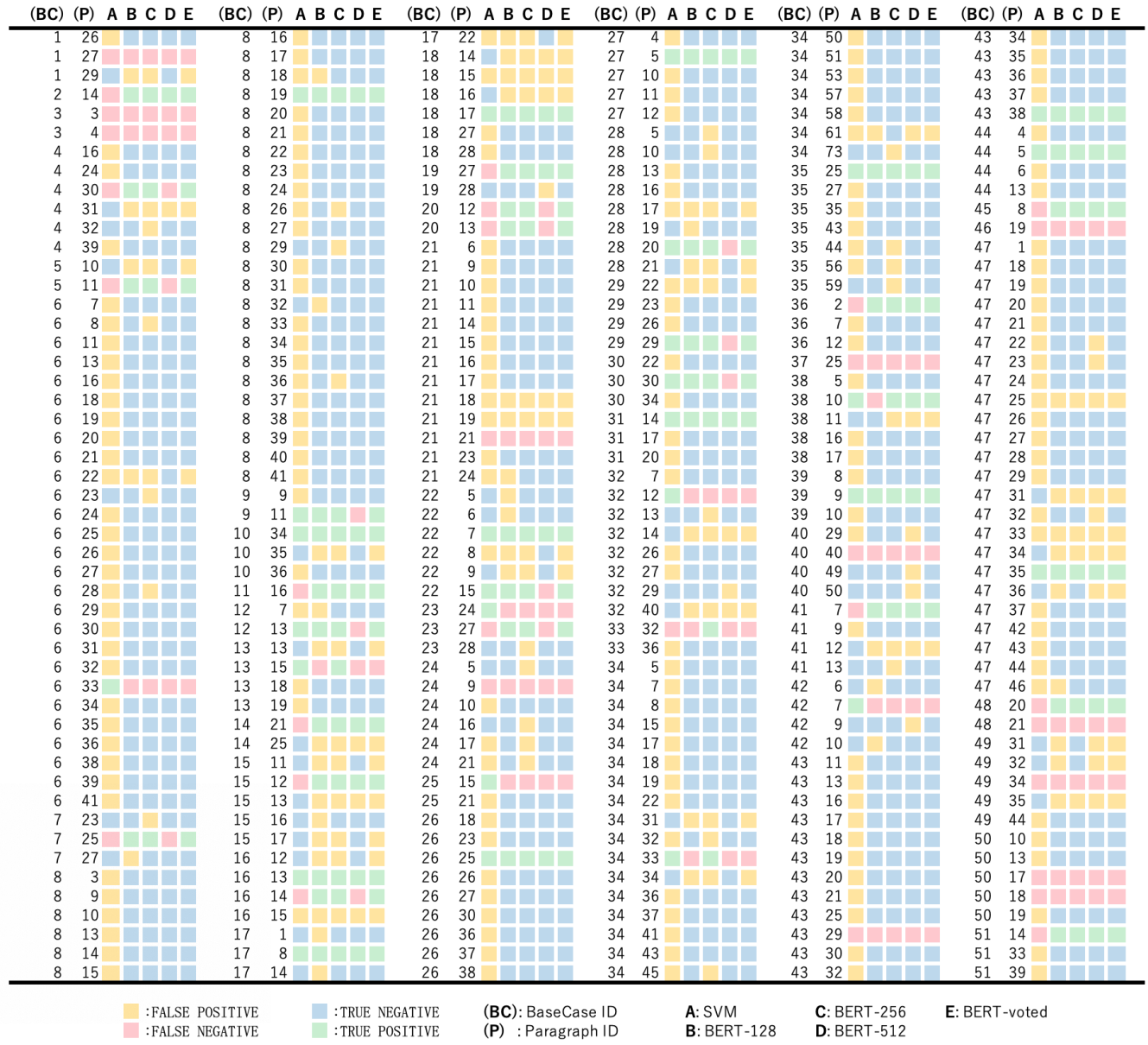


Figure 4: Visualized results of cross validation with training data (First 51 cases)

Table 6: Results of formal run

Models	Precision	Recall	F1
BERT-256	0.40	0.80	**0.53**
BERT-512	0.38	0.69	0.49
BERT-vote	0.39	0.73	0.51

5 CONCLUSION

In this paper, we proposed an SVM-based baseline model and four BERT-based models and compared their performance on COLIEE task 2. The SVM-based model adopted a linear kernel and inputs by the universal sentence encoder. The BERT-based models were pre-trained with the out-domain data (Wikipedia corpus and Book corpus) and fine-tuned with the COLIEE task 2 data. According to our dry run experiments, the BERT models performed significantly better than the baseline model and the ensemble model was the best among the four BERT-based models, achieving $F = 0.50$. The score is significantly improved from $F = 0.26$, which was the best score in the last competition although the data set is different. We successfully fitted the model to the legal textual entailment task by fine-tuning without any feature engineering. This result shows

Entailed fragment

Only where unreasonable omissions have been made, such as the failure to investigate obviously crucial evidence, is judicial review warranted

Entailing paragraph (correct answer, 021.txt)

[21] With respect to judicial review of decisions of the Canadian Human Rights Commission, Jerome, A.C.J., held in Lukian v. Canadian National Railway Co. (1994), 80 F.T.R. 38 (T.D.):

"Generally, when Courts are called upon to review the exercise of an administrative tribunal's discretionary power, they will be reluctant to interfere since tribunals, by virtue of their training, experience, knowledge and expertise, are better suited than the judiciary to exercise those powers. Provided the Commission's decision is within the discretion given to it, the Court will not interfere with the manner in which it was exercised, unless it can be shown the discretion was exercised contrary to law. What the law requires is the Commission to consider each individual case before it, to act in good faith, to have regard to all relevant considerations and not be swayed by irrelevant ones, and to refrain from acting for a purpose contrary to the spirit of its enabling legislation or in an arbitrary or capricious manner."

Figure 5: Hard case sample correct answer (train-21)

Entailing paragraphs (predicted by BERT-vote, 018.txt, 019.txt)

[18] In Slattery, supra, at 20 to 23, Nadon, J., required the elements of neutrality and thoroughness to be present in the Commission's investigation as these elements are necessary to provide a fair basis on which the Commission evaluates whether a tribunal should be appointed pursuant to s. 44(3)(a) of the Act. With regard to neutrality, it has been held that if the Commission simply adopts an investigator's conclusions without giving reasons, and those conclusions were made in a biased manner, a reviewable error occurs: Canadian Broadcasting Corp. v. Canadian Human Rights Commission et al. (1993), 71 F.T.R. 214 (T.D.). The requirement of thoroughness of investigation is discussed in Slattery and stems from the essential role that investigators play in determining the merits of particular complaints. In determining the degree of thoroughness of investigation required to be in accordance with the rules of procedural fairness, the interests of the complainant and the respondent must be balanced with the Commission's interest in maintaining an administratively workable system. Deference must be given to administrative decision-makers to assess the probative value of evidence and to decide whether to further investigate accordingly. **Only where unreasonable omissions have been made, such as the failure to investigate crucial evidence, is judicial review warranted.** In instances where parties have the legal right to make submissions in response to an investigator's report, parties may be able to compensate for minor omissions by bringing them to the attention of the decision-maker. Therefore, only where complainants are unable to rectify such omissions should judicial review be considered.

[19] In Slattery, supra, at 28, Nadon, J., set up a high threshold for judicial review of Commission decisions on the ground of thoroughness: "The fact that the investigator did not interview each and every witness that the applicant would have liked her to and the fact that the conclusion reached by the investigator did not address each and every alleged incident of discrimination are not in and of themselves fatal as well. This is particularly the case where the applicant has the opportunity to fill in gaps left by the investigator in subsequent submissions of her own. In the absence of guiding regulations, the investigators, much like the CHRC, must be master of its own procedure, and judicial review of an allegedly deficient investigation should only be warranted where the investigation is clearly deficient."

Figure 6: Hard case sample predictions (train-21)

the possibility of the fine-tuning with large pre-trained language models in the legal text entailment task. Although the BERT-based models showed good results, there is room for further improvement. We observed some cases where the models recognized non-entailing sentences as entailing due to the surface level matching. Introducing the domain knowledge and the appropriate representation of legal augmentations might remedy such errors. Furthermore, a separate feature extractor and a bypassed network for the legal dedicated knowledge could help to solve the problem.

REFERENCES

[1] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. *Journal of machine learning research* 3, Feb (2003), 1137–1155.

[2] Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. 2018. Universal Sentence Encoder. *CoRR* abs/1803.11175 (2018). arXiv:1803.11175 http://arxiv.org/abs/1803.11175

[3] Chih-Chung Chang and Chih-Jen Lin. 2011. LIBSVM: A library for support vector machines. *ACM transactions on intelligent systems and technology (TIST)* 2, 3 (2011), 27.

[4] Corinna Cortes and Vladimir Vapnik. 1995. Support-vector networks. *Machine Learning* 20, 3 (01 Sep 1995), 273–297. https://doi.org/10.1007/BF00994018

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[6] Mohit Iyyer, Varun Manjunatha, Jordan Boyd-Graber, and Hal Daumé III. 2015. Deep unordered composition rivals syntactic methods for text classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, Vol. 1. 1681–1691.

[7] Mi-Young Kim, Yao Lu, Juliano Rabelo, and Randy Goebel. 2018. COLIEE-2018: Evaluation of the Competition on Case Law Information Extraction and Entailment. In *Proceedings of the Twelfth International Workshop on Juris-informatics (JURISIN 2018)*. 105–116.

[8] Edward Loper and Steven Bird. 2002. NLTK: The Natural Language Toolkit. In *Proceedings of the ACL-02 Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics - Volume 1 (ETMTNLP '02)*. Association for Computational Linguistics, Stroudsburg, PA, USA, 63–70. https://doi.org/10.3115/1118108.1118117

[9] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).

[10] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. In *Advances in Neural Information Processing Systems 26*, C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger (Eds.). Curran Associates, Inc., 3111–3119. http://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf

[11] COLIEE 2019 organizing committee. 2019. COLIEE-2019. https://sites.ualberta.ca/~rabelo/COLIEE2019/. Accessed: 2019-04-10.

[12] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.

[13] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 1532–1543.

[14] Juliano Rabelo, Mi-Young Kim, Housam Babiker, Randy Goebel, and Nawshad Farruque. 2018. Legal Information Extraction and Entailment for Statute Law and Case Law. In *Proceedings of the Twelfth International Workshop on Juris-informatics (JURISIN 2018)*. 142–155.

[15] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. *Technical report* (2018).

[16] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.). Curran Associates, Inc., 5998–6008. http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf

[17] Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*. 19–27.

HUKB at the COLIEE 2019 Information Retrieval Task - Utilization of metadata for relevant case retrieval -

Masaharu Yoshioka
yoshioka@ist.hokudai.ac.jp
Faculty of Information Science and Technology, and Global Station for Big Data and Cybersecurity, Global Institution for Collaborative Research and Education, Hokkaido University
Sapporo-shi, Hokkaido, Japan

Zihao Song
Graduate School of Information Science and Technology, Hokkaido University
Sapporo-shi, Hokkaido, Japan

ABSTRACT

HUKB participated in two information retrieval tasks of COLIEE 2019 (Task 1 for legal case retrieval and Task 3 for statute law retrieval). We introduce our systems for the tasks and discuss the effectiveness of the systems based on the evaluation results.

CCS CONCEPTS

• **Information systems** → **Structured text search**; *Query reformulation*; *Retrieval effectiveness*.

KEYWORDS

Information retrieval, legal case retrieval, metadata

ACM Reference Format:
Masaharu Yoshioka and Zihao Song. 2019. HUKB at the COLIEE 2019 Information Retrieval Task - Utilization of metadata for relevant case retrieval - . In *Proceedings of COLIEE 2019 workshop: Competition on Legal Information Extraction/Entailment (COLIEE 2019).* ACM, New York, NY, USA, 5 pages.

1 INTRODUCTION

Competition on Legal Information Extraction/Entailment (COLIEE) 2019 has four tasks as a combination of document domains (legal cases and statute law articles) and task types (information retrieval and entailment). We (HUKB) participated in two information retrieval tasks: legal case retrieval (Task 1) and statute law retrieval (Task 3).

In this paper, we introduce our system for the submission of final results. For Task 1, we use metadata information extracted from the legal case documents. For Task 3, we use almost the same settings as those used for COLIEE 2018 [6].

2 LEGAL CASE RETRIEVAL TASK

Task 1 is a legal case retrieval task that is designed to find supporting legal cases for the decision of new legal case Q from the legal case corpus[1]. For each query, the supporting cases are selected from noticed cases of the query case.

We first review our previous system used at COLIEE 2018 and propose a method to utilize metadata of the legal case documents. We also discuss the effectiveness of the system based on the evaluation results of COLIEE 2019.

2.1 Legal Case Retrieval System at COLIEE 2018

Legal case documents have common structures. However, the document data provided by the organizers are not well formalized. In the previous COLIEE, we implemented the system to extract structural information based on pattern matching of a section, such as lines in the text, and to classify texts into three parts [6].

Yoon v. Can. (M.C.I.) (2012), 405 F.T.R. 139 (FC)
MLB headnote and full text
– snip –
Indexed As: Yoon et al. v. Canada
(Minister of Citizenship and Immigration)
Federal Court
Shore, J.
February 9, 2012.
– snip –
Summary:

The Refugee Protection Division (RPD) ...
– snip –
Aliens - Topic 1329.3
– snip –
[1]
Shore, J.
[Translation]: The RPD occupies ...
– snip –

Figure 1: Examples of Legal Case Documents

- Header parts
  Header parts contain a variety of information, such as reference information, name of the court, date, summary, counsel, and solicitors of record. We extract the

[1] https://sites.ualberta.ca/ rabelo/COLIEE2019/

COLIEE 2019, June 21, 2019, Montreal, Quebec

summary information by checking the line that starts with `Summary:`. Following lines just after `Summary:` are lines with other heading keywords (e.g., `Administrative Law`, `Statues Noticed`, and `Counsel`) that comprise summary sections. If the system fails to extract summary parts from the header, the system uses all header text of a legal case as a summary.

- Fact parts:
  Several documents present the fact part in two languages (English and French). In most cases, the French parts are followed by English and the same numbers are used. Thus, we exclude text parts after the appearance of the second language `[1]`.
- Footer parts:
  We do not use footer parts for the experiment.

By using this simple text extraction framework, a summary and a fact part are extracted from each candidate legal case document.

In COLIEE 2018, queries were given as summary part texts and fact part texts. From the preliminary experiments, we decided to use summary text only for the query and use summary and fact texts for constructing the document database of the legal case documents.

Another item discussed in the previous COLIEE 2018 system is a method to construct a legal case document database for calculating statistical information of the term frequency. Because all test data are distributed as a pair of query and candidate legal case documents, there are two simple methods for constructing the document database.

- DB_{each}: The database for the IR system was constructed by using only the candidate legal case documents (we constructed a database for each query).
- DB_{all}: The database for the IR system was constructed by using all legal case documents. For training analysis, we used documents for training data only, but for the final submission, we used all documents (training and test) for database construction. However, retrieved legal cases from other databases are excluded from the final submission results.

In the preliminary experiments, it was better to use DB_{all} because DB_{all} has larger numbers and variations of documents. Construction of the database with a limited number of candidate legal cases for one query may cause bias when estimating the statistical information of the term frequency in all the legal case documents to be examined. For example, if the number of relevant cases is larger than 10 and all relevant cases contain a characteristic term that is important when checking for similarity, when we use DB_{each}, the system calculates that more than 5% of documents contain this keyword as document frequency information. In such a case, the IR system that uses document frequency for estimating the importance of a term cannot appropriately estimate the importance of the keyword. Therefore, we use DB_{all} for all experiments.

We use Indri [4] [2] as the basic information retrieval tool and select the relevant legal cases based on the ranks of the retrieved cases of DB_{all}.

2.2 Utilization of metadata of the legal case documents

As explained in Section 2.1, legal case documents consist of multiple parts (summary, facts, date, name of the court, citation, topics, and others). In Task 1 of COLIEE 2018, no participants used metadata (such as date, name of the court, or topics) of the legal case documents for retrieving the relevant legal cases.

In this paper, we propose to use two types of metadata (date and topics) for the retrieval task. The idea of utilizing metadata is very simple. For the date, we compared the date of noticed legal cases and the target legal cases and confirmed that there are no referred legal cases whose date is newer than the target case; there are several examples that consider the legal cases whose date is the same as that of the target case. It is obvious that it is impossible to notice a future legal case.

In addition, most of the questions have topic metadata that represent key issues to be discussed in legal cases. Therefore, when legal cases share the same topic, it may be an important clue for identifying the relevant legal cases.

We utilize such metadata information to extend our previous IR system for the legal case retrieval tasks explained in Section 2.1.

2.3 System architecture

In addition to extracting the summary and fact parts from the legal case, we extract two additional metadata items from the header part of legal case documents.

Date when the trial was concluded: We extract the line that contains the month, date, and year from the header parts. For example, in the header part of the base case for training data 001 (Figure 1), we extract date information from a line such as "February 9, 2012".

Topics related to the case: In the header section, topics relevant to the case are listed with the topic number. For example, in the header part of the base case for training data 001, we extract date information from lines that contain "Topic" with numbers (such as "Aliens - Topic 1329.3" shown in Figure 1)). From this legal case document, we extract five topics: 1329.3, 4083, 2745, 2807, and 8309.

We combine features extracted from the metadata with a previous system using LinearSVM of scikit-learn[3] to determine whether the candidate document is relevant.

The following are features for the SVM.

- Number of shared topics
  The system compares the topics of the query (base case) and the topics of the candidate legal case and

[2] https://www.lemurproject.org/indri/
[3] https://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVC.html

counts the number. Because there are relevant legal cases that are used for discussing a part of the issue in the base case, we only use a number of shared topics instead of normalizing the value (i.e., using a ratio of shared topics or cosine similarity).

- Normalized Indri score for using the summary as a query
  This Indri score is the same as that used in COLIEE 2018 (Section 2.1). Because the Indri score is represented as a logarithm of probability for the relevance, the scores are represented as negative numbers. We normalize the Indri score of *did* for query *qid* ($Indri_{qid,did}$) using the maximum Indri score of query *qid* (Max_{qid}) using the following formulas.

$$Max_{qid} = \max Indri_{qid,did} | did \in Doc_{qid}$$
$$Norm_{qid,did} = 10^{(Indri_{qid,did} - Max_{qid})}$$

  $Norm_{qid,did} = 1$ for the maximum score documents of *qid*, and the other scores are below 1.
- Normalized Indri score for using summary and fact as a query
  In addition to the score based on the summary, we use Indri scores calculated by using summary and fact as a query. We apply the same normalization procedure for the summary as a query to the scores.

Date metadata are used for the pre-processing to select candidate legal case documents.

Based on the preliminary experiments, we found that LinearSVM with out-of-the-box settings tend to generate smaller numbers of relevant cases because of the imbalance between the number of relevant cases and other cases. For the worst case, the system selects no relevant cases from the candidate cases.

To manage this unbalanced situation, we decided to use the following settings for SVM training.

- Exclusion of legal cases with newer dates than the base case
  In order to reduce the number of irrelevant cases, we exclude legal cases with newer dates than the base case.
- Parameter tuning for `class_weight`
  LinearSVM module of scikit-learn has parameter `class_weight` for representing the weight of each class to make better-balanced results.
- Introduction of a minimum number (R_{min}) of relevant case selections
  To avoid a situation where there are no relevant legal cases for the base case, we introduce a minimum number of relevant cases. If the number of selected relevant cases is lower than this number, the system selects additional legal cases based on the order of the `decision_function` value that calculates the distance to the separate hyperplane.

Table 1: Evaluation results of the submitted runs for each team (Task 1)

run	precision	recall	F
HUKB	0.7021	0.4000	0.5097
JNLP	0.6000	0.5545	0.5764
ILPS	0.6810	0.4333	0.5296
IITP	0.6368	0.3879	0.4821
CLArg	0.9266	0.3061	0.4601
UA	0.3560	0.3333	0.3443
HUKB-topic	0.3323	0.3303	0.3313
CACJ	0.2119	0.5848	0.3110

Table 2: Macro average of HUKB and HUKB-topic

run	precision	recall	F
HUKB	0.6627	0.5464	0.5291
HUKB-topic	0.5613	0.5040	0.4343

To calculate the appropriate value for `class_weight` and R_{min}, we conducted 10 cross-fold validation tests using different parameter settings and decided to use `class_weight` = 7 and $R_{min} = 2$.

We also conducted additional experiments without using shared topic information for discussing the effectiveness of using shared topic metadata (HUKB-topic).

Table 1 shows the evaluation results of the submitted run and best runs submitted by other teams.

HUKB ranks third out of seven submissions. Comparing the previous COLIEE 2018 results and this year's results, the difference between the top-ranked team and ours (F-value of JNLP was 0.6546 and ours was 0.3808 at COLIEE 2018[2]) was significantly reduced. From this comparison, we confirmed that our proposed method is effective for the task.

From the comparison with HUKB-topic, we confirmed that using shared topic metadata is effective. One of the characteristics of HUKB-topic is generating larger numbers of relevant cases for one query. For example, HUKB-topic selects 48 candidate cases for query 17 that has only two noticed cases. To avoid the effects of errors by selecting a larger number of relevant cases for one query, we calculated the evaluation measure using the macro average (average of evaluation measures for each query) (Table 2).

Compared with micro average evaluation results, the difference between HUKB and HUKB-topic has decreased. However, HUKB is still better than HUKB-topic in all evaluation measures. From these results, we confirmed that using the topic metadata is useful for discriminating whether the cases are noticed and for finding cases with less similarity from the perspective of the IR system.

Because a simple selection of candidate documents based on the date information can be easily applied to other systems, it may be necessary to consider this effect when selecting candidate legal cases for each base legal case.

3 STATUTE LAW RETRIEVAL TASK

3.1 System architecture

For Task 3 (Statute Law Information Retrieval Task), we use almost the same system for the COLIEE 2018 that was developed based on the system's previous COLIEE tasks [3, 5].

The following is the basic system architecture of our previous system [6].

(1) Extract condition and decision parts from the query and articles
We manually extract the conditions and decisions of the articles and compare the conditions and decisions of the queries that are extracted automatically [6] using dependency parser KNP [4].
(2) Calculate similarity using Indri
We constructed three types of queries (conditions, decisions, all text). We also constructed four document databases that contain a part of a document (condition, decision, title, and contents) and one document database with all text (title and contents). We calculated Indri-based similarity scores for combinations of all queries and all document databases except for the condition query with the decision database and the decision query with the condition database using Indri [4] ($3 \times 5 - 2 = 13$ similarity scores are calculated).
(3) Calculate the longest common subsequence with various pairs of queries and text parts
For the same 13 variations introduced, scores based on the longest common subsequence proposed in [5] were calculated.

As a result, 26 feature values were collected for each candidate article. All feature values were normalized to the 0–1 scale using maximum and minimum values using linear transformation (maximum values correspond to 1 and minimum values correspond to 0).

We use the information as features for SVM rank [1] to calculate the similarity scores. We use the SVM rank software [5] based on scikit-learn[6]. To manage the varieties of the query types, we set up SVM rank models with different training sets and merged the results. In this year, we constructed 12 variations of models for SVM rank by excluding one-year data (H19–H29, H18+H20–H29, ..., and H18–H28) for training data. When the new query is given, the system calculates ranks of each article by using these SVM rank models and aggregate results by calculating the average ranks of each article. Lastly, the top-ranked article is a candidate article.

3.2 Submission results

We submitted one result for Task 3. However, after the first evaluation was conducted, we found that there is a bug for normalizing the score. Consequently, we conducted additional experiments using a system without the normalization bug (HUKB-revised).

Table 3 shows the evaluation results of the submitted runs and the best runs of other teams. Columns "run", "LANG", and "F2" correspond to the name of the team (best runs for each team are selected), language used for making the database, and F-2 value (harmonic means of precision and recall with more weight to recall), respectively. Columns "MAP" and "Recall@5" correspond to mean average precision calculated by ranked results and recall calculated by the top five ranked documents, respectively.

The original submission of HUKB is the worst performance in F2 measure. After removing the bugs, the results improved slightly, but they were not as good as other submissions from other teams. Thus, it is necessary to conduct failure analysis to investigate why the new submission is not as good as the other teams' submissions.

However, our revised system is as good as "Recall@5", suggesting that there are the following two discussion issues for our system.

(1) Our system is not good at selecting one appropriate article from the highly ranked candidates.
(2) Our system has the potential to find relevant articles for the question with multiple relevant articles, but the system returns only one article for each question.

We plan to investigate the system by considering those aspects.

4 SUMMARY

In this paper, we introduced our system for participating in Tasks 1 and 3 of COLIEE 2019. For the Task 1 system, we found that using metadata for the legal case helps identify relevant cases. For Task 3, because of the time limitation for preparing the system, the same system as that used for COLIEE 2018 is used with different training data. However, the performance of the system is worse than the scores submitted by other teams for COLIEE 2018. Thus, it is necessary to investigate the causes for the lower performance score by using detailed failure analysis.

ACKNOWLEDGMENT

This work was partially supported by JSPS KAKENHI Grant Number 16H01756 and 18H0333808.

REFERENCES

[1] Thorsten Joachims. 2006. Training Linear SVMs in Linear Time. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '06)*. ACM, New York, NY, USA, 217–226. https://doi.org/10.1145/1150402.1150429
[2] Mi-Young Kim, Yao Lu, Juliano Rabelo, and Randy Goebel. 2018. COLIEE-2018: Evaluation of the Competition on Case Law Information Extraction and Entailment. In *The Proceedings of the 12th International Workshop on Juris-Informatics (JURISIN2018)*. The Japanese Society of Artificial Intelligence,, 105–116.
[3] Daiki Onodera and Masaharu Yoshioka. 2016. Civil Code Article Information Retrieval System based on Legal Terminology

[4] http://nlp.ist.i.kyoto-u.ac.jp/EN/index.php?KNP
[5] https://gist.github.com/agramfort/2071994
[6] http://scikit-learn.org/

Table 3: Evaluation results of submitted runs (Task 3)

run	LANG	F2	precision	recall	MAP	Recall@5
HUKB	J	0.4140	0.4490	0.4102	0.4935	0.4876
HUKB-revised	J	0.4542	0.5000	0.4493	0.5827	**0.6777**
UA	E	**0.5493**	0.5918	0.5442	**0.6181**	0.6198
JNLP	E	0.5343	0.4592	0.5820	0.5982	0.6529
EVORA	E	0.5334	**0.5714**	0.5289	0.6275	0.6694
JNLP	E	0.5054	0.4031	0.5616	0.5750	0.5950
KIS	J	0.5032	0.4227	**0.6126**	0.5625	0.6281
DBSE	E	0.4659	0.4544	0.4932	0.5119	0.5124
iitp	E	0.4472	0.4898	0.4422	0.5409	0.5702

and Civil Code Article Structure. In *The Proceedings of the 10th International Workshop on Juris-Informatics (JURISIN2016)*. Paper 19.

[4] Trevor Strohman, Donald Metzler, Howard Turtle, and W Bruce Croft. 2005. Indri: A language model-based search engine for complex queries. In *Proceedings of the International Conference on Intelligent Analysis*. 2–6.

[5] Masaharu Yoshioka and Daiki Onodera. 2017. A Civil Code Article Information Retrieval System based on Phrase Alignment with Article Structure Analysis and Ensemble Approach. In *COLIEE 2017. 4th Competition on Legal Information Extraction and Entailment (EPiC Series in Computing)*, Ken Satoh, Mi-Young Kim, Yoshinobu Kano, Randy Goebel, and Tiago Oliveira (Eds.), Vol. 47. EasyChair, 9–22.

[6] Masaharu Yoshioka and Zihao Song. 2018. HUKB at COLIEE2018 Information Retrieval Task. In *The Proceedings of the 12th International Workshop on Juris-Informatics (JURISIN2018)*. The Japanese Society of Artificial Intelligence,, 183–193.

Question Answering System for Legal Bar Examination using Predicate Argument Structures focusing on Exceptions

Reina Hoshino
Graduate School of Science and Technology
Shizuoka University
Hamamatsu, Shizuoka, Japan
rhoshino@kanolab.net

Naoki Kiyota
Department of Socio-Information Studies
Shizuoka University
Hamamatsu, Shizuoka, Japan
nkiyota@kanolab.net

Yoshinobu Kano
Graduate School of Science and Technology
Shizuoka University
Hamamatsu, Shizuoka, Japan
kano@inf.shizuoka.ac.jp

ABSTRACT

We developed a textual entailment system for Japanese legal bar exam, which can explain the way system solves based on underlying logical structures. We focus on the condition clauses that had been a problem in our previous year's challenge. We added a new function that handles the exception conditions. We made a variation of different modules using different sentence comparison methods based on predicate argument structures. We used SVM to make ensemble of these modules. In addition, we made a legal synonym dictionary, which is structured in that synonym of a predicate could be conditioned with its object. We applied our system to the COLIEE 2019 Task 4 dataset. Our system achieved the second-best rank, 0.62 of accuracy score.

CCS CONCEPTS

• Natural Language Processing → Lexical semantics; Language resources.

KEYWORDS

COLIEE, Question Answering, Legal Bar Exam, Legal Information Extraction, Predicate Argument Structure Analysis.

ACM Reference format:
Reina Hoshino, Naoki Kiyota and Yoshinobu Kano. 2019. Question Answering System for Legal Bar Examination using Predicate Argument Structures focusing on Exceptions, In Proceedings of COLIEE 2019 workshop, ICAIL 2019, Montreal, Canada

1 Introduction

The COLIEE shared task series is held in association with the JURISIN (Juris-informatics) workshop. The first one was the COLIEE 2014 shared task[1]. Following this, COLIEE 2015 shared task[2], COLIEE2016 shared task[3], COLIEE 2017[4], and COLIEE 2018[5] shared task (this time in conjunction with ICAIL) were held. The COLIEE shared task consists of four tasks. We challenged Task 4. Task one and Task 2 used the legal cases. Task 1 of this legal question answering task involves reading a legal bar exam question and extracting a subset of Japanese Civil Law Articles. Task 3 is given a problem, and searches articles necessary to solve the problem. Task 4 is a problem and related article and answer it with Yes/No whether the content is correct. In last year's, it was necessary for our system to search for articles necessary to solve the problem without being given related articles. The previous Task 4 was a combination of Task 3 and Task 4 this time. By eliminating the need to search related articles, we can confirm more reading comprehension and logical judgment skills.

Regarding question answering in general, most successful systems employ rather traditional techniques of question answering which have decades of history [6]–[9]. Recently, automatic answering system used machine learning by big data. The typical example is IBM Watson System [10]. The core Watson system employed a couple of open source libraries, including the traditionally well-designed DeepQA system [11] as its skeleton of question answering processing.

However, there are only hundreds of problem sentences used in COLIEE. We think that the amount of data is not enough to perform an end-to-end supervised machine learning. There are an enormous number of legal documents, many methods have been proposed to acquire knowledge from case examples [12]. However, it is difficult to use case examples directly in the legal bar exam, because we need commonsense knowledge to follow the thinking of a legal expert, not just the explicit knowledge base available. In addition, a question answering system for legal documents is required to show reasons of its decision in a human interpretable way. Our aim is to create a legal question answering system where we can trace evidences of it decision.

In this paper, we describe our automatic problem solver for the legal bar exam that captures structures of legal documents and understands their meaning. We focus on Task 4 throughout this paper. Our main achievements in

COLIEE 2019, June 21, 2019, Montreal, Quebec

our previous COLIEE 2018 participation were our synonym dictionary and manual classification analysis of human capabilities, which are required to solve each problem. We added a couple of new features based on our COLIEE 2018 system, showing that correct answer rates were improved. Our system achieved the second-best score among participants in Task 4, which accuracy score was 0.62.

2 System Architecture

2.1 System Overview

We designed our system based on our previous system in COLIEE 2018. Because Task 4 of COLIEE 2019 is a textual entailment task where the target gold standard articles are given, we do not use our related article search function in our previous system. Article search was required in Task 4 of COLIEE 2018 as it was a question answering task without any target article given.

A civil law article consists of a legal effect part and a cases part to state its effect. If an event is not applicable to the cases, the corresponding legal effect does not state. We divide a given sentence into a case part and an effect part, in order to express the relationship between the effect and the cases. We perform this division based on predicate argument structure analysis.

We classified past problems in terms of human capabilities required to solve the given problem. This analysis showed what is missing in our previous solver system. One of our classification kinds is a problem type that requires detailed sentence analysis, where we need to extract conditional clauses (or sentences) in problems and related articles. Although this conditional clause extraction is a very common issue in the legal bar exam, our correct answer rate was not sufficient for this problem type. While our previous system extracts the conditional part in a sentence, which should have not been sufficient.

Because there are exceptional conditions e.g. “Except …”, we added a new function in order for our system to extract the exceptional conditional part. When an exceptional condition is true, the effect may not be exerted even if the main conditions are true. Furthermore, additional conditions exist for some cases, when the exceptional condition is true. As our previous system does not handle such exceptional conditions, we added a new function to extract such exceptional conditions. A civil law article could be multiple sentences, which structure has rules as follows. In most cases, the first sentence describes a condition and the effect when that condition is met. In the second and subsequent sentences, an exception of the first sentence’s condition may be mentioned. For example, “ただし、…しない。(However, … not…)” shows that exceptional conditional sentences follow in this sentence. Even if the conditions described in the first sentence are met, this exceptional condition could invert the boolean value of the entire proposition. Our new function handles this sort of exceptions

We analyzed predicate argument structures that are used in legal documents, adding new entries to the synonym dictionary. Different predicates could be used to express the same meaning. Humans could make use of analogy to resolve such paraphrases, but we need explicit knowledge for systems. We defined our structured synonym dictionary to compare pairs of predicates between problems and articles, checking the given pair has the same meaning or not. In this comparison, we compare predicates and objects using our structured synonym dictionary. We also try to use FrameNet[13], which is a semantic frame database, to determine synonyms.

We used our own structured synonym dictionary, which holds relationships of pairs of predicate and object. Entries of our structured synonym dictionary could be regarded to have the same content. We enhanced this dictionary from our previous system.

Our system internally consists of multiple modules. Each module outputs answers individually, then our system integrates these answers into a final answer. We tried different ways of the integration as described later.

2.2 Sentence Preprocessing

In an itemized article, effect is described first, then cases are listed to show the effect. In such a case, we need the entire article information rather than individual sentence (or item). Case list tends to omit subjects or predicates, which makes the linguistic post-processing more difficult.

As a countermeasure, we concatenate the case list with the first sentence. As the first sentence always contains a phrase “次に掲げる (the following things)”, we delete this specific phrase and then concatenate the case list. We repeat this process for the number of items, generating a sentence for each item. Unfortunately, concatenated sentences are sometimes syntactically invalid, causing errors in predicate argument structure analysis.

Text with parenthesis is another problem. Some parentheses contain punctuation marks. For example, consider the following case: “縁側（ベランダを含む、次項において同じ、）を設ける者は、目隠しを付けなければならない。(A person who installs a window or porch (including a verandah. Hereinafter in this and the following paragraph) must put up a privacy screen.)” In this case, a sentence is inserted within the parenthesis, but the whole sentence makes sense both with the outer and inserted inner sentence. Texts in the parentheses sometimes indicate exceptional conditions, while sometimes may simply be a supplementary explanation, which are difficult to distinguish automatically. In our system, we regard texts in parenthesis as one of condition clauses of the entire sentence, by deleting the parenthesis in a sentence and replacing punctuation marks if any, to be a single sentence.

When a given problem consists of two or more sentences, we regard a proposition clause of the latter sentence as a proposition

clause of the entire problem, and all other clauses as condition clauses of the entire problem. This is because former sentences tend to explain situations and personal attributes; latter sentences are more important. This type of multiple sentence problems was first presented in COLIEE 2018.

2.3 Morphological Analysis and Case Analysis

We used the morphological analyzer JUMAN[14] and the case analyzer KNP[15]. We added our own term dictionary for the legal domain: legal vocabulary dictionary[16], terms extracted from the PROLEG [17] source code, and terms added manually as follows. This dictionary is same as our previous system.

In order to create the term list manually, we examined vocabulary from past eleven years of the COLIEE problems. We found that more than 300 kinds of words are used in problems per one year. More than 50 new words appeared per year when registering words from older year to newer year (Table 1). We added 200 new words from these extracted words by manually selecting which word to register.

We added other words as well, extracting from source codes of PROLEG[17]. PROLEG is a logic programming language for the legal domain based on Prolog. They systemize internal structure of the law from the civil law articles and actual cases. Function names and variable names of PROLEG are named by hand. We added these names to our morphological analyzer's dictionary, when they consist of Japanese letters only. These names are necessary when reading and understanding the legal documents deeply.

Table 1. Number of words used in the COLIEE training data (previous years' problems)

	total	new
H18(2006)	351	-
H19(2007)	317	175
H20(2008)	322	144
H21(2009)	402	171
H22(2010)	316	87
H23(2011)	312	73
H24(2012)	484	127
H25(2013)	425	50
H26(2014)	563	144
H27(2015)	490	84
H28(2016)	441	67

2.4 Predicate Argument Structure Analysis, Condition Clause and Proposition Clause

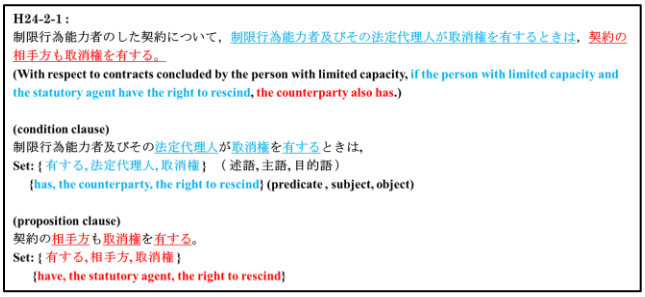


Fig. 1. An example of predicate argument structure analysis

We used predicate argument structure based on our previous system. Fig. 1 shows an example of the predicate argument structure analysis. We defined our own original clause unit ("節") in order to recognize condition clauses and main clauses precisely, which are included in a single sentence. A clause should include a single predicate as a core element of that clause. We apply a dependency parser that makes chunks ("文節") of a couple of morphemes. Starting form a chunk that includes a predicate, we aggregate neighboring chunks when a neighboring chunk does not include any predicate, until a clause unit is formed.

A predicate is not always suitable to be a core predicate of a clause. For example, "holding" in "condition holding a court" could be regarded as a predicate. However, this is not suitable as a core single predicate in a clause because we need to compare larger predicate-argument structures rather than such a noun phrase.

We define two types of special clauses: a proposition clause and a condition clause. A proposition clause includes an end of the sentence. In the Japanese language, a clause which includes an end of sentence often represents a proposition. We regard a clause as a condition clause when that clause includes specific patterns, e.g." when…"," in case of …", etc.

For each sentence, we compare proposition clauses of the problem sentences and the civil law articles using these sets. The same applies for condition clauses. We use the base form of predicates for their comparison. For example, "認める (admit)" and "認めない (do not admit)" have the same meaning, sharing the same base form "認める (admit)".

Normally, a sentence consists of a proposition clause and an arbitrary number of condition clauses. Because an article consists of one or more sentences, we compare sentences one by one when comparing the article sentences with the problem sentence.

2.5 Exception Conditions

Article 9 :
成年被後見人の法律行為は、取り消すことができる。ただし、日用品の購入その他日常生活に関する行為については、この限りでない。
(A juristic act performed by an adult ward may be rescinded; provided, however, that, this shall not apply to any act relating to daily life, such as the purchase of daily household items.)

Article 32 (2):
踪の宣告によって財産を得た者は、その取消しによって権利を失う。
ただし、現に利益を受けている限度においてのみ、その財産を返還する義務を負う。
(Any person who acquired any property by the adjudication of disappearance shall lose its/his/her right upon rescission thereof; provided, however, that such person shall have the obligation to return such property only to the extent he/she is actually enriched.

Fig. 2 An example of exception conditions

Our system classifies each sentence into three types: "not exception", "positive exception condition", and "negative exception condition" using "ただし、…しない。(However, … not…)" that negates the preceding sentence. For example, "前文は適用されない(the previous sentence does not apply)" is a "negative exception condition", while "当てはまる場合は、…する(If the condition is met, then…)" is a "positive exception condition". We need this distinction because negation could make the answer reversed.

The above classification is performed as follows. If a sentence starts with "ただし、…(However, …)", it is one of exception condition types. Then if the sentence ends with "…ない(not)", it is regarded as "negative exception condition". This classification uses simply string match as the COLIEE sentences are very uniformly written. There could be, however, other style of exception expressions which cannot be handled by our current system. We classify a sentence but not an entire article into these types, because sometime only one of article's sentences is an exception condition.

For a sentence of exception condition, our system compares all of clauses in the exception condition sentence with condition sentences in the problem. If this comparison does not match, our system uses the comparison result of the previous sentence. If this comparison matches, and if this is the negative exception condition, we revert the boolean value of our answer. If this comparison matches, but if this is the positive exception condition, we compare the exception condition sentence and problem sentence. These matches are performed by comparing the predicate argument structures as described before.

2.6 Synonym Dictionary for Predicates

t1 :
第三百五十六条　不動産質権者は、質権の目的である不動産の用法に従い、その使用及び収益をすることができる。
(Pledgees of immovable property may use and receive the profits from the immovable property that is the subject matter of a pledge, in accordance with the method of its use.)

t2 :
不動産質権者は，質権の目的物である不動産の用法に従いこれを使用することができるが，不動産から生じた果実を取得することはできない。
(A pledgee of immovable property may use and receive obtain fruits from the immovable property that is the subject matter of a pledge, in accordance with the method of its use, but may not collect fruits derived by the real property.)

t1: 収益する (receive the profits)
t2: 取得する (obtain) - condition: 果実 (fruits)

Fig. 3. An example of synonym dictionary

We manually created our structured synonym dictionary for predicates. For example, "成立する (effect)" and "適用する (apply)" could have the same meaning. Our dictionary is structured with object conditions. For example, "果実を取得する (obtain fruits)" and "収益する (receive the profits)" have the same meaning when they have the same object. We referred to the past problems of 2009 to 2006 to make our dictionary. We compared related articles and corresponding problem sentences to manually extract our synonym dictionary entries. As a result, we extracted 33 combinations of verbs.

2.7 Problem Solver Modules

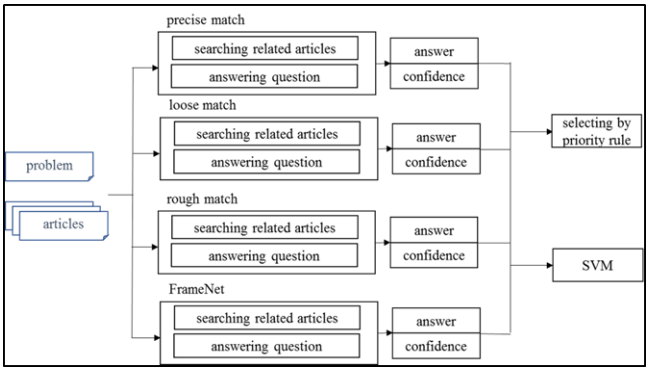


Fig. 4 Overview of our module integration

We implemented four types of problem solver modules: precise match, loose match, rough match, and FrameNet modules. Our system outputs the final answer by integrating outputs of these different modules.

For each predicate, the precise match module finds a subject and an object by their case markers. We perform exact matches for the predicate, its subject and its object between a pair of problem and a given article. When we could not find any subject nor any object, we skip that sentence. If a negative expression is detected only in one side, we reverse the corresponding Yes/No

answer. Our system repeats these processes for the number of the clauses of the problem sentences.

Loose match module is a looser version of the precise match. When comparing proposition clauses and condition clauses, this module regards a pair of clauses as matched if either a pair of objects or a pair of subjects is matched, in addition to a pair of predicates.

For example, if a problem's clause is {結ぶ (sign), 代理人 (agent), 契約(contract)} and if an article's clause is {結ぶ (sign), 代理人 (agent), 売買契約 (sales contract)}, then our precise match module judges that they do not have a same meaning because they do not have a same object. On the other hand, our loose match module judges that they have a same meaning as this module ignores objects in this case.

Rough match module is the loosest match in our modules. This module only compares predicates of proposition clauses and then checks whether the predicates include negative expressions or not.

Furthermore, we prepared the FrameNet module. FrameNet is an English semantic lexical database based on a theory of meaning called Frame Semantics. The basic idea is that people understand the meaning of words largely by the frames which they evoke. Frames have some semantic roles called Frame Elements (FEs) and frame evoking words are called Lexical Units (LUs). We use LUs of Japanese FrameNet, in addition to the original English version of FrameNet. We use Japanese WordNet[18], which is a lexical database for Japanese. We use Japanese WordNet to obtain synonyms, hypernyms, hyponyms and English translation words to expand FrameNet. The FrameNet module is made by extending the loose match module. The difference of the FrameNet module from the loose match module is in comparing predicates. Using an inheritance relationship of FrameNet, we examine whether a pair of predicates represents the same meaning or not. This system measures a distance between nodes of words and regards as the same meaning if the nodes are in a near distance. We defined confidence values for each relationship type to calculate the distance.

In addition, we made more variations of modules by setting different options for these four modules. For example, as a new feature from the previous year's system, the precise match module was added a new variation that handles the exception conditions. After completing the answers for each module, we decide our final answer. We prepared two ways for the module integration as follow.

In the first way of module integration, we try applying the precise match module first. When the precise match module cannot be applied, we apply the loose match module. If the loose match module cannot be applied as well, we try applying the rough match module.

We use SVM (Support Vector Machine) for the second way of module integration, We use each answer output by each module and the confidence calculated from the number of articles that matched word sets of the problem sentence for learning. The confidence is a value that decreases as more the number of the related articles. The equation of calculating the confidence is 1 / n. n is the number of the related articles when n > 0. The confidence is 0 when n = 0. However, the related articles of solving the problems are not only one, it is not necessarily true that the probability that the related articles is less and accurate is higher. The training data for SVM is the problems of 2006 to 2014. The optimum cost and gamma value of SVM options were determined by grid search. We used the polynomial kernel.

3 Experiments and Results

We applied our system to Task 4 of COLIEE 2019. In Task 4, the problem sentences and the gold standard related articles were provided without any additional information, asking Yes/No as answers.

3.1 Result of COLIEE 2019

Table 2. Result of COLIEE 2019 formal run

Team	# Correct Answers (total 98 answers)	Accuracy
BaseLine	51	0.5204
UA_Ex	67	0.6837
KIS_3module	61	0.6224
IITP	58	0.5918
KIS_dic	58	0.5918
UA_Go	58	0.5918
KIS_frame	57	0.5816
DBSE	56	0.5714
JNLP.t=98	56	0.5714
TRAttn	55	0.5612
TRSimFeat	52	0.5306
JNLP.t=85	51	0.5204
EVORA1	50	0.5102
JNLP.t=78	48	0.4898
EVORA3	47	0.4796
EVORA2	44	0.4490

Table 2 shows the result of the COLIEE 2019 Task 4 formal run. Among these results, team names with KIS as their prefix show our results.

KIS_3module uses the precise match module, the loose match module and the rough match module. This system selected answer by the priority. KIS_dic used our synonym dictionary. The difference from KIS_3module is not only synonym dictionary, but this system uses SVM for the module integration. Confituration of KIS_dic is close to combining KIS_dict and KIS_SVM of COLIEE 2018. KIS_frame uses the FrameNet module only. KIS_3module was the best result among our submissions for the

Table 3. Evaluation results with textual entailment tasks of COLIEE training data (past years's legal bar exam)

	2006	2007	2008	2009	2010	2011	2012	2013	2014	total
KIS_3module	0.60	0.52	0.58	0.68	0.60	0.63	0.56	0.59	0.57	0.591
KIS_dic	0.54	0.50	0.58	0.52	0.60	0.54	0.63	0.68	0.56	0.579
KIS_frame	0.57	0.55	0.62	0.64	0.49	0.61	0.58	0.67	0.53	0.583
KIS_SVM (baseline)	0.54	0.57	0.58	0.63	0.51	0.63	0.54	0.61	0.59	0.577

COLIEE 2019 formal run. Our system achieved the second-best score in Task 4.

Table 3 shows results of COLIEE training data of past years' legal bar exam. We did not use data after 2015 because the original official data includes irregular formats. The total number of questions is 504.

Baseline is the result of KIS_SVM, which performed the best among our submissions in the previous COLIEE 2018. Baseline results are calculated using the same module configuration as KIS_SVM, where the gold standard related articles are used.

Overall, KIS_3 module performed the best, which also performed the best in the formal run, which should be the most effective combination of modules. On the contrary, KIS_dic and KIS_frame was in the reverse order of formal run. However, this is not always true for the results of individual year, suggesting that there would be different tendency of problems, as a different module combination sometimes performs better.

4 Discussion

At the time of the preivous COLIEE 2018 Task 4, KIS_SVM results were the best in our development, and KIS_Frame using FrameNet was the best in the fomal run, so we configured KIS_dic and KIS_frame similar to them in this year. However, KIS_3module was the best in this year's formal run, probably because the configuration of KIS_3module would have largely effected by the new function of exception condition sentences. Our new function of exception conditions should have contributed to increase the correct answer rate. As the previous COLIEE 2018 Task 4 was not textual entailment but question answering that requires searching articles, different functions may be required depending on task differences, such as word disambiguation by FrameNet.

All of the results were better than the baseline, suggesting that our pre-processing which is used for all of modules has contributed to the scores. Our pre-processing could have simplified the complicated structured sentences without losing their meanings.

Our score still has a difference of couple of points from the first-ranked team. There could be several future works to improve our system. For example, condition clauses are not sufficiently handled yet while condition clauses very often appear. Logical inference over multiple sentences is another issue. We converted multiple sentences into a single sentence by our pre-processing. While we can capture legal document structure to some extent in this way, we ignore information that can only be expressed by the original multiple sentences. As human experts can understand documents very precisely word by word.

3.2 Results evaluated by training data

5 Conclusion and Future Work

We developed an automatic problem solver system for legal bar exam, which can show logical structures how the system solved a given legal problem.

In order to represent structures of legal documents, we implemented a textual entailment solver, which has predicate argument structure analysis as its core part.

We focus on the condition clauses that had been a problem in our previous year's challenge. We added a new function, that handles the exception conditions.

We made a variation of different modules using different sentence comparison methods. We used SVM to make ensemble of these modules. In addition, we made a legal synonym dictionary. Our synonym dictionary is structured in that synonym of a predicate could be conditioned with its object. We enhanced this dictionary.

We applied our system to the COLIEE 2019 Task 4 dataset. Our system achieved the second-best rank, 0.62 accuracy score .

ACKNOWLEDGMENTS

This work was partially supported by MEXT Kakenhi and JST CREST.

REFERENCES

[1] "Competition on Legal Information Extraction/Entailment (COLIEE-14), Workshop on Juris-informatics (JURISIN) 2014," 2014. [Online]. Available: http://webdocs.cs.ualberta.ca/~miyoung2/jurisin_task/index.html.

[2] M.-Y. Kim, R. Goebel, and S. Ken, "COLIEE-2015: Evaluation of Legal Question Answering," in *Ninth*

International Workshop on Juris-informatics (JURISIN 2015), 2015.

[3] M. Kim, R. Goebel, Y. Kano, and K. Satoh, "COLIEE-2016 : Evaluation of the Competition on Legal Information Extraction and Entailment 2 . The Legal Question Answering Task," 2016.

[4] Y. Kano, M. Kim, R. Goebel, and K. Satoh, "Overview of COLIEE 2017," vol. 47, no. Icail, pp. 1–8, 2017.

[5] M. Yoshioka, Y. Kano, N. Kiyota, and K. Satoh, "'Overview of Japanese Statute Law Retrieval and Entailment Task at COLIEE-2018,'" *Twelfth Int. Work. Juris-informatics (JURISIN 2018)*, pp. 1–12, 2018.

[6] D. Lin and P. Pantel, "Discovery of Inference Rules for Question Answering," *Nat. Lang. Eng.*, vol. 7, no. 4, pp. 343–360, 2001.

[7] D. Pinto, A. McCallum, X. Wei, and W. B. Croft, "Table Extraction Using Conditional Random Fields," *Proc. 26th Annu. Int. ACM SIGIR Conf. Res. Dev. informaion Retr. - SIGIR '03*, p. 235, 2003.

[8] D. Ravichandran and E. Hovy, "Learning Surface Text Oatterns for a Question Answering System," *Proc. 40th Annu. Meet. Assoc. Comput. Linguist. - ACL '02*, no. July, p. 41, 2001.

[9] H. Yu and V. Hatzivassiloglou, "Towards Answering Opinion Questions: Separating Facts from Opinions and Identifying the Polarity of Opinion Sentences," *Proc. 2003 Conf. Empir. methods Nat. Lang. Process.*, pp. 129–136., 2003.

[10] D. A. Ferrucci, "Introduction to 'This is Watson,'" *IBM J. Res. Dev.*, vol. 56, no. 3.4, pp. 1:1-1:15, 2012.

[11] D. A. Ferrucci, "IBM's Watson/DeepQA," *SIGARCH Comput. Arch. News*, vol. 39(3), no. Journal Article. doi:10.1145/2024723.2019525, 2011.

[12] S. Polsley, P. Jhunjhunwala, and R. Huang, "CaseSummarizer: A System for Automated Summarization of Legal Texts," *Proc. COLING 2016, 26th Int. Conf. Comput. Linguist. Syst. Demonstr.*, pp. 258–262, 2016.

[13] "Welcome to FrameNet!" [Online]. Available: https://framenet.icsi.berkeley.edu/fndrupal/.

[14] "日本語形態素解析システム JUMAN."(Japanese morphological analysis system JUMAN) [Online]. Available: http://nlp.ist.i.kyoto-u.ac.jp/index.php?JUMAN.

[15] "日本語構文・格・照応解析システム KNP."(Japanese syntax / case / reference resolution analysis system KNP) [Online]. Available: http://nlp.ist.i.kyoto-u.ac.jp/index.php?KNP.

[16] 法令用語研究会, 有斐閣 法律用語辞典(Yuhikaku legal term dictionary), 第 4 版(fourth edition). 有斐閣, 2012.

[17] K. Satoh, K. Asai, and T. Hurukawa, "PROLEG : PROLEG : Implementing the Presupposed Ultimate Fact Theory of Japanese Civil Code by Logic Programming," 知識ベースシステム研究会 / 人工知能学会 (Knowledge-Based Systems / The Japanese Society for Artificial Intelligence), vol. 92, pp. 1–8, 2011.

[18] "Japanese WordNet." [Online]. Available: http://compling.hss.ntu.edu.sg/wnja/index.en.html.

IITP in COLIEE@ICAIL 2019: Legal Information Retrieval using BM25 and BERT

Baban Gain
gainbaban@gmail.com
Government College of Engineering and Textile Technology, Berhampore
Berhampore, West Bengal

Dibyanayan Bandyopadhyay
dibyanayan@gmail.com
Government College of Engineering and Textile Technology, Berhampore
Berhampore, West Bengal

Tanik Saikh
1821cs08@iitp.ac.in
Indian Institute of Technology, Patna
Bihta, Patna, Bihar

Asif Ekbal
asif@iitp.ac.in
Indian Institute of Technology, Patna
Bihta, Patna, Bihar

ABSTRACT

Natural Language Processing (NLP) and Information Retrieval (IR) in the judicial domain is an essential task. With the advent of availability domain-specific data in electronic form and aid of different Artificial intelligence (AI) technologies, automated language processing becomes more comfortable, and hence it becomes feasible for researchers and developers to provide various automated tools to the legal community to reduce human burden. The *Competition on Legal Information Extraction/Entailment (COLIEE-2019)* run in association with the *International Conference on Artificial Intelligence and Law (ICAIL)-2019* has come up with few challenging tasks. The shared defined four sub-tasks (i.e. Task1, Task2, Task3 and Task4), which will be able to provide few automated systems to the judicial system. The paper presents our working note on the experiments carried out as a part of our participation in all the sub-tasks defined in this shared task. We make use of different Information Retrieval(IR) and deep learning based approaches to tackle these problems. We obtain encouraging results in all these four sub-tasks.

CCS CONCEPTS

• **Term Frequency - Inverse Document Frequency**; • **Doc2Vec**; • **BM25**; • **BERT**; • **Entailment**;

KEYWORDS

BM25, tf-idf, text similarity, Entailment

ACM Reference Format:
Baban Gain, Dibyanayan Bandyopadhyay, Tanik Saikh, and Asif Ekbal. 2019. IITP in COLIEE@ICAIL 2019: Legal Information Retrieval using BM25 and BERT. In *Proceedings of COLIEE 2019 workshop: Competition on Legal Information Extraction/Entailment (COLIEE 2019).* ACM, New York, NY, USA, 5 pages.

COLIEE 2019, June 21, 2019, Montreal, Quebec

1 INTRODUCTION

Language technology based automated system has a great impact and demand in judicial domain. As for example, it will mitigate the problem of manual checking of large amount of previous files to find relevancy to a current case. This kind of system would be very effective to lawyers which will expedite the process of providing verdict on a particular case. Previously, it took couple of months and/or even a year to get verdict of a particular case. It is observed that there are many cases where practitioners rely on previous court case files to extract information and try to find relevancy with him/her current solving case. In normal practise they do it process manually. However, computerised processing of such judicial domain texts is taxing as the files are very large in size and remain unstructured. This stipulates lot of research in this domain. The *Competition on Legal Information Extraction and Entailment (COLIEE-2019)* an associated event of *(ICAIL)-2019* has defined four problems related to IR in this domain. Task 1 and Task 2 are involved the case law competition, Task 1 and Task 2 are associated with statute law competition. The data for these two types of tasks are extracted accordingly. We participate in all the four tasks (i.e. Tasks 1, Tasks 2, Tasks 3 and Tasks 4) defined in this competition. All of these four tasks are vary emerging and are of utmost priority.

Task 1: This task is basically legal case retrieval task. Overall the task is: after reading an new case Q, the system has to extract supporting cases *S1, S2, S3, ...Sn* from the provided case law dataset. This will support the decision for Q.
Task 2: This legal case entailment task, which involves the detection of a paragraph from the existing cases that entails the decision of a new case.
Task 3: The task 3 is essentially a legal Question-Answering task. The task could be defined as: reading a legal bar examination question Q and extracting a subset of Japanese Civil Code Articles *S1, S2,..., Sn* from the entire Civil Code which are those appropriate for answering the question such that i.e.
Entails(S1, S2, ..., Sn , Q) or Entails(S1, S2, ..., Sn , not Q).
Overall, the task is given a question Q and the entire Civil Code Articles, the system has to retrieve the set of articles *"S1, S2, ..., Sn"* as the answer of this track.
Task 4: This task involves legal question answering task which associates the identification of an entailment relationship such that

Entails(S1, S2, ..., Sn , Q) or Entails(S1, S2, ..., Sn , not Q).
Given a question *Q*, the participating systems have to retrieve relevant articles *S1, S2, ..., Sn* in the first phase, and in the second phase, the systems have to determine if the relevant articles entail "Q" or "not Q". The answer of this track would be binary: i.e. *"YES"("Q") or "NO"("not Q")*.
We propose various approaches to combat these problems. We discuss all our approaches in the section 4.

2 OUR APPROACHES

The propose approaches for the all four tasks are based on *Doc2Vec*, *BM25*, *tf-idf*, *Bidirectional Encoder Representations from Transformers (BERT)* [2]. The following subsections will discuss all the approaches one by one.

2.1 Doc2Vec

Doc2Vec [3] is an unsupervised algorithm that learns fixed-length feature representations from variable-length pieces of texts, such as sentences, paragraphs, and even a whole documents. This algorithm represents each document by a dense vector which is trained to predict words in the document. More precisely, we concatenate on the paragraph vector with several word vectors from a paragraph and predict the following word in the given context. Both word vectors and paragraph vectors are trained by the stochastic gradient descent and back-propagation [6]. While paragraph vectors are unique among paragraphs, the word vectors are shared. At prediction time, the paragraph vectors are inferred by fixing the word vectors and training the new paragraph vector until convergence.

2.2 BM25

BM25 [5] is one of the most successful text-retrieval algorithm. BM25 is a bag-of-words retrieval function that ranks a set of documents based on the query terms appearing in each document, regardless of the inter-relationship between the query terms within a document. Given a query Q, containing keywords q1,q2,....qn the BM25 score of a document D is:

$$Score(D,Q) = \sum_{i=1}^{n} IDF(q_i) \frac{f(q_i,D).(k_1+1)}{f(q_i,D) + k_i.(1-b+b.\frac{|D|}{avgdl})} \quad (1)$$

where f(q_i, D) is q_i's term frequency in the document D, |D| is the length of the document D in words, and *avgdl* is the average document length in the text collection from which documents are drawn.

2.3 Term Frequency * Inverse Document Frequency

*Term Frequency and Inverse Document Frequency (tf*idf)* is a numerical statistic that is intended to reflect how important a word is to a document in a collection of corpus. It is often used as a weighting factor in searches of information retrieval, text mining, and user modeling. The tf*idf value increases proportionally to the number of times a word appears in the document and is offset by the number of documents in the corpus that contain the word, which helps to adjust for the fact that some words appear more frequently in general. The formula that is used to compute the tf*idf for a term t of a document d in a document set is:

$$tf - idf(t,d) = tf(t,d) * idf(t) \quad (2)$$

We implement this using python library [4].

2.4 Bidirectional Encoder Representations from Transformers (BERT)

We make use of a novel language representation model called *Bidirectional Encoder Representations from Transformers (BERT)*. The model is the latest language representation model used in various tasks of NLP. The task 4 defined in this shared task is a classification task. We treat the task as a sentence classification task by considering 't1' and 't2' as two sentences respectively. Using BERT we obtain a vector corresponding to the '[CLS]' token at the last layer. This vector is subsequently used for classification. So per 't1' and 't2' we get one vector to be subsequently used in a downstream model (e.g a feed-forward neural net layer) for further classification. We tried with adding 1 or 2 Dense layers but actually the accuracy obtained was not good to report. The vector obtained was of 768 dimensions and the training set contains 724 examples. Each example contains more dimensions of a vector than the number of examples combined. So the data is prone to be over-fitted. So we resort to use a popular machine learning model known as *eXtreme Gradient Boosting (XGBoost)* [1] for classification. XGBoost gained much popularity recently as a winning solution in several machine learning competitions. It is actually a library which provides gradient boosting framework in several languages. We treat each of the dimension of the vector corresponding to '[CLS]' as a separate feature. So we got 724 training examples each with 768 features. We use a matrix representing this data with dimension [724,768]. This is given to the XGBoost model for classification.

3 DATA

The whole competition task is divided into two categories *viz 1. Case Law Competition (Task 1 and task 2) 2. Statute Law Competition (Task 3 and Task 4)*. The task organizer's released the data for all these four tasks defined. The data is obtained from an existing collection of predominantly Federal Court of Canada case law. In task-1 the training data consists of 285 case queries and each query has 200 candidate notice cases. So each query has 9.87 notice cases on average. The training data of task-2 comprises of 181 queries and each query has an average of 48.59 candidate paragraphs for recognizing entailment relation. In the training data of this task on an average 1.32 paragraphs have an entailment relation with a query. For the task 3 and 4 data are borrowed from Japanese civil law articles (translated to English) and the size of the data for these two tasks are same having 1044 articles.

4 SYSTEM DESCRIPTION

There were four sub-tasks in this competition. We attempt all of those. In the following subsections we discussed about the experimental procedures, models etc. Basically, the pre-processing module for task 1 and task 2 is the same, which is described in the following paragraph.

Pre-processing: The pre-processing module comprises of two stages. In the first stage, i.e. for the first model (named as IITPd2v),

we perform minimal amount of pre-processing. All the candidate cases and the base cases are extracted and the paragraph numbers are removed. In the second stage i.e. for second model (named as iitpBM25), we remove all the words having length lower than 3 character and all the numerical values from every documents. We also remove all the stop-words using NLTK tool. Then we lemmatize the words using NLTK WordNetLemmatizer [1].

4.1 Task1

The goal of this task is to explore and evaluate case law retrieval technologies that are both effective and reliable. The task investigates the performance of systems that search a set of legal cases that support a previously unseen case description. The input of this system is a query and return noticed cases in the given collection as output. We say a case is 'noticed' with respect to a query iff the case supports the decision of the query case. In this task, the query case does not include a decision, because our goal is to determine how accurately a machine can capture decision-supporting cases for a new case (with no decision).
The training data consists of 285 numbers case queries for this task, and each query has 200 numbers candidate noticed cases. In the training data of this task, each case query has an average of 9.87 noticed cases. We propose three models for this task namely *IITP Doc2Vec (IITPd2v)*, *iitpBM25* and *iitpDocBM*

4.1.1 Models. We have three models to tackle this problem, we will discuss all these in detail in the following paragraphs.
iitpd2v: In this model, we train the data obtained from stage-1 of pre-processing module. For training purpose we make use of gensim Doc2Vec with vector size = 150, window = 10, and with 50 epochs. For every query, we compute the Doc2Vec similarity between the base case and all of its candidates. As the task contains variable number of noticed cases, top 10 candidates whose similarity was greater than 90% of average score of top two are returned. Total 245 documents are returned for 61 base cases. This model yields the precision, recall, and F-measure as 0.4653, 0.3455, and 0.3965 respectively.
iitpBM25: From this model, we collect the data yielded by the stage-2 of pre-processing module. We compute BM25 score of all the candidate cases for every base case using gensim library. Then the top 10 candidates whose similarity is greater than 90% of average score of top two are returned. Thus, the total of 203 documents are returned. This model produces the precision, recall, and F-measure as 0.6256, 0.3848, and 0.4765 respectively.
iitpDocBM: In this model we combine the scores obtained from the previous two models i.e. from the iitpd2v and from the iitpBM25. The make multiplication of those two scores, so that only documents with high scores on both Doc2Vec and BM25 are ranked. Top 10 candidates whose similarity was greater than 80% of average score of top two are returned. In this way total of 201 documents are returned. We compute the precision, recall, and F-measure and get the scores as 0.6368, 0.3879, and 0.4821 respectively for this model.

[1] https://pythonprogramming.net/lemmatizing-nltk-tutorial/

4.2 Task2

In this task, the goal is to predict the decision of a new case by entailment from previous relevant cases. As a simpler version of predicting a decision, a decision of a new case and a noticed case will be given as a query. Then a case law textual entailment system must identify which paragraph in the noticed case entails the decision, by comparing the extracting and the meanings of the query and paragraph.

4.2.1 Model. We propose three models for this task. The name of the models are *viz. i. iitp2D2v ii. iitpBM25 and iii. iitp2DocBM*
iitp2D2v: For iitp2D2v module, we take the data from stage-1 of pre-processing module which we train using gensim Doc2Vec. The specifications for the training is as follows: vector size = 100, window = 5, and epochs = 50. For every query, Doc2Vec similarity between the entailed fragment and all of its candidates are computed and the candidate with highest score are returned. This yields the Precision, Recall, and F-measure of 0.0455, 0.0444, and 0.0449 respectively.
iitpBM25:For iitpBM25, we take the data from stage-2 of the pre-processing module. For every entailed fragment BM25 score of all the candidates are computed by employing gensim library. Then candidate with highest score are returned. This system yields the Precision, Recall, F-measure scores of 0.7045, 0.6889, and 0.6966 respectively. Please note that we got the highest Precision score among all the participants in this task. We obtain the F-score which is the second highest among unique teams.
iitp2DocBM: In iitp2DocBM module, the scores obtained from iitp2D2v and iitpBM25 are multiplied, so that only candidates with high scores on both the methods are ranked. Then the candidate with the highest score is returned. We obtain the Precision, Recall, and F-measure score as 0.6591, 0.6444, and 0.6517 respectively in this task.

4.3 Task3

This task investigates the performance of the systems that search a static set of civil law articles using previously unseen queries. The goal of this task is to return relevant articles in a collection to a query. We call an article as "Relevant" to a query iff the query sentence can be answered as *Yes/No*, i.e. entailed from the meaning of the article. If combining the meanings of more than one article (e.g., "A", "B", and "C") can be answered a query sentence, then all the articles ("A", "B", and "C") are considered as "Relevant". If a query can be answered by an article "D", and it can also be answered by another article "E" independently, we also consider both of these articles "D" and "E" as "Relevant". This task requires the retrieval of all the articles that are relevant to answering a query.

4.3.1 Model. We propose four models in this task namely *iitpBM25, iitpBM25-L, iitptfidf and iitptfidf-L*. The following points will discuss all the models in detail.
iitpBM25: In this module, for every query BM25 score of all the articles are computed using gensim library. Then candidates having the highest score are returned. The total 98 results are retrieved among which 54 are correct. Total number of correct entries in the

corpus was 121. We obtain the Precision, Recall, and F2-measure of 0.5510, 0.4462, and 0.4639 respectively.

iitpBM25-L: This model is the modified version of the previous one. We extract the top 100 documents in Descending Order of their score for each query. 109 out of 121 articles in the corpus are retrieved correctly. ***iitptfidf:*** In this model, for every query tf-idf score of all the articles are computed using sklearn TfIdfVectorizer library[2]. The candidates having the highest scores are returned. Total 98 articles are retrieved among which 49 are correct. Total number of correct entries in the corpus was 121. So the precision, recall, and F2-measure provided by this system are 0.50, 0.4049, and 0.4209 respectively.

iitptfidf-L: This model also the modified version of the previous one. We retrieve top 100 documents for each query. The model retrieves 108 articles out of which 121 articles correct.

4.4 Task4

Pre-processing: In the Task 4 the pre-processing procedure is as follows. We extract the data from the given .xml files containing tags named as 'pairs', 't1', and 't2'. Under a 'pair' of 't1' and 't2', we have to find their entailment relationship. We use the 'id' attribute under the 'pair' tag as a key to store the corresponding article(t1) and the document(t2). For the training of the model we make use of the training data provided for Task 3 and tested the system using the test data provided in the Task 4. We extract the task-4 data for training in similar ways as of task-3 discussed above. Here along with 't1' and 't2', we also got the 'label' attribute under pair tag which are used as training labels. We gather all 't1's and 't2's along with their labels to form the training dataset. In Task 4 for training purpose we got 12 files containing 724 training examples with 353 instances as *'Yes'* and 371 instances *'No'*.

4.4.1 Model. Here only one model is offered. The description of the model is as follows.

iitpbert: In this task we propose a model named as *'iitpbert'*. We make use of BERT-Base-Uncased (Bidirectional Encoder Representation from Transformer)[2] model to combat this problem. We provide all the 't1's and 't2's as inputs to the BERT model. This task is basically a sentence pair classification task. So we obtain the final hidden state (i.e. output of Transformer) corresponding to the first *'[CLS]'* token as the vector representation for the classification tasks. We check that the combined length of t1 and t2 should be less than 512 or not. BERT model can handle the maximum length of 512. We strip the tokens after index 512, if it exceeds 512. We obtain a matrix of shape N*M where N is number of training examples and M is the dimension of the output vector of BERT (here M = 768). We try to add dense layers and other machine learning models like SVM and XgBoost on top of that output vectors to train them end to end. This yields an accuracy of around 55% when validated on a held out train set. We choose to stay with the trained XgBoost classifier and use it to predict labels for the task-4. We submit that file with predicted file for evaluation. This approach produces an accuracy of 59.18% on the test set for this task.

Table 1: Results obtained by proposed models in Task-1

Models	Precision	Recall	F-measure
iitpd2v	0.4653	0.3455	0.3965
iitpBM25	0.6256	0.3848	0.4765
iitpDocBM	0.6368	0.3879	0.4821

Table 2: Results obtained by proposed models in Task-2

Models	Precision	Recall	F-measure
iitpd2v	0.0455	0.0444	0.0449
iitpBM25	0.7045	0.6889	0.6966
iitpDocBM	0.6591	0.6444	0.6517

Table 3: Results obtained by four proposed models in Task-3

Models	Precision	Recall	F2-measure	MAP
iitpBM25	0.4898	0.3967	0.4142	X
iitpBM25-L	X	X	X	0.5409
iitptfidf	0.4388	0.3554	0.3694	X
iitptfidf-L	X	X	X	0.5056

5 RESULTS AND DISCUSSIONS

The task organizer's defined various measures to evaluate the participating systems. The task 1 and task 2 rely on precision, recall and F-measure. In task 3, in addition to precision, recall and F2-measure they also encourage the participant to take Mean Average precision (MAP) into account. Whereas, in task 4 only accuracy is provided as the evaluation metric. The results for the task1, and task-2 in three methods are shown in the Table 1 and Table 2 respectively. The results for the task-3 using four models are shown in the Table 3

Please note that here we have calculated Mean Average Precision (MAP) by taking top 100 ranked documents into account. The MAP scores are shown in iitpBM25-L and iitptfidf-L methods in the 5th column of the Table 3. In task 4 participants are encouraged to compute the accuracy to evaluate their submitted systems. So in this task we compute the accuracy on from our *Yes/No* prediction. We obtain the accuracy of 59.18% for the task with our proposed model.

6 CONCLUSION AND FUTURE WORK

The paper presents our experiment's report performed as a part of our participation in all the four sub-tasks defined in the shared task entitled *COLIEE-2019* organized in *ICAIL 2019*. We propose various text retrieval methods (Doc2Vec, BM25, and tf-idf) for sub-task 1, sub-task 2, and sub-task 3. BERT model is employed for tackling the sub-task 4. We obtain encouraging results in all the four sub-tasks and our ranks in all the four task are in a quite modest position among all the participants in the leaderboard. Our future line of research could be something as follows:

- Using some good pre-trained vector embedding with sent2vec could produce better result, so we would link to foster this in future.

[2]https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html

- We would like to use BERT-large model (another version of BERT) to explore how it is performing.
- We would like to do retrospective analysis of the best three models in this competition to get some insights and for better improvement further.

7 ACKNOWLEDGEMENTS

Mr Tanik Saikh acknowledges all the co-authors for their contributing efforts. He is also acknowledging the funding agency to provide the fund for the registration (if anything needs to be done).

REFERENCES

[1] Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*. ACM, 785–794.

[2] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *CoRR* abs/1810.04805 (2018). arXiv:1810.04805 http://arxiv.org/abs/1810.04805

[3] Quoc V. Le and Tomas Mikolov. 2014. Distributed Representations of Sentences and Documents. In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*. 1188–1196. http://jmlr.org/proceedings/papers/v32/le14.html

[4] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in Python. *Journal of machine learning research* 12, Oct (2011), 2825–2830.

[5] Stephen Robertson, Hugo Zaragoza, et al. 2009. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends® in Information Retrieval* 3, 4 (2009), 333–389.

[6] David E Rumelhart, Geoffrey E Hinton, Ronald J Williams, et al. 1988. Learning representations by back-propagating errors. *Cognitive modeling* 5, 3 (1988), 1.

Retrieving Legal Cases with Vector Representations of Text

Guilherme Paulino-Passos
g.passos18@imperial.ac.uk
Department of Computing
Imperial College London
United Kingdom

Francesca Toni
f.toni@imperial.ac.uk
Department of Computing
Imperial College London
United Kingdom

ABSTRACT

Task 1 of the COLIEE competition proposes a challenge in information retrieval of legal cases: given a legal decision from Canadian courts, with references to previous cases omitted, the task is to find which cases from a pre-selected pool were actually cited by the decision. This task captures important aspects of the challenge of developing methods for finding relevant cases for supporting, for instance, a practising lawyer. In this work, we present an approach to address this challenge based on vector representation of cases. Vector representations of text have been successfuly used for a variety of natural language processing tasks, such as semantic similarity and textual entailment. We explore the performance of some such methods with out-of-the-box tools, in combination with two different classifiers: random forests and k-nearest neighbours. Additionaly, we explore the impact of adding a feature consisting of how many times a particular past case has appeared in the entire dataset. We find that this feature increases the F1 score of at least 19.89 percent points. We hypothesise this may be an artifact of the way the dataset is constructed. With the inclusion of this feature, in our experimental setup, we obtain results of 62.10% F1 score, in cross-validation evaluation.

CCS CONCEPTS

• **Information systems** → **Information retrieval**; • **Applied computing** → **Law**; • **Computing methodologies** → *Natural language processing*; Machine learning.

KEYWORDS

information retrieval, case law, document embeddings, vector representations, classification

ACM Reference Format:
Guilherme Paulino-Passos and Francesca Toni. 2019. Retrieving Legal Cases with Vector Representations of Text. In *Proceedings of COLIEE 2019 workshop: Competition on Legal Information Extraction/Entailment (COLIEE 2019).* ACM, New York, NY, USA, 6 pages.

1 INTRODUCTION

Legal information retrieval has practical importance, as much of lawyers' activities is finding documents relevant to a new case at hand. Examples are applicable statutes, academic opinions, and similar past cases. These documents can be persuasive as well as authoritative sources of law, and are therefore crucial for supporting legal argumentation [Schauer 2009, pp. 61-75].

One interesting way of drawing the attention of the academic community to a particular problem is by the creation of shared tasks (or challenges), in which a problem is transformed into a controlled, well-defined version, with clear evaluation criteria. Participants then submit systems which try to solve the problem, which are then ranked. The COLIEE (Competition On Legal Information Extraction and Entailment) is one example, calling for participation in four different tasks, amounting to information retrieval and textual entailment tasks in two different scenarios: case law and statute law.

Legal systems across the world are usually roughly divided into two archetypes: common law systems and civil law systems [Schauer 2009; Tetley 2000].[1] In the archetype of common law systems, legal content is mostly created from legal decisions, by a mechanism of respecting the precedent called *stare decisis*. Roughly speaking, according to this, whenever a legal problem is given a specific legal solution, it is mandatory to apply the same solution for future cases with the same legal problem, under similar circumstances. In this way, the content of law is made by the decisions of judges. Contrary to this, in the archetype of civil law systems, legal content is typically created by legislators (e.g. a parliament), who give rise to general and abstract rules (usually in specific texts called *codes*) that should then be applied by the courts [Schauer 2009, pp. 103-108]. Even if real legal systems do not exactly follow such descriptions, these are informative about the main focus and attitude from participants in particular systems. However, it has been claimed that many legal systems in the world are incorporating features of both, that is, both systems are approximations. That is, precedents are becoming more important in civil law countries, while in common law countries the creation of detailed statutes and legal interpretation techniques are being emphasized [Schauer 2009, p. 108]. Thus, research on intelligent systems for case law, such as developing methods for finding related past cases, arguments on precedents, and legal content defined on legal decisions, can be useful not only for traditionally considered common law legal systems, but for civil law systems as well.

COLIEE 2019, June 21, 2019, Montreal, Quebec

[1] Those names are originated from two particular traditions in the West: the English tradition and the Roman one. The former includes the United Kingdom, the United States, Australia, Canada, India, among others; while the latter includes continental Europe and Latin America, among others. However, here we are following Schauer [2009, pp. 103-104] in considering two stereotypical descriptions of the approach to law. For instance, for these purposes Schauer [2009] considers Islamic law, Chinese law, and socialist law as civil law traditions.

We present an approach for COLIEE Task 1: information retrieval on case law. A summary of the task is: given a legal decision from Canadian courts, with references to previous cases omitted, the task is to find which cases from a pre-selected list were actually cited by the decision. This simulates a situation in which there is a new situation at hand, and a lawyer would like to find past cases which could be informative about the law regarding the new case.

In this work, we use an approach based on vector representation of texts [Manning et al. 2008, pp. 107-123]. In particular, we use vectors derived from term frequency and inverse document frequency (tf-idf) and from sentence embeddings generated from entailment data [Cer et al. 2018a,b; Conneau et al. 2017; Manning et al. 2008], analysing whether this kind of sentence embedding provides improvements over regular tf-idf. We treat the problem as a classification problem using features derived from such representation of data. We compare the results of two different classifiers: random forests and k-nearest neighbours. Additionally, we investigate the impact on performance of the models of adding a feature consisting of how many times a particular past case has appeared in the entire dataset. We find that this feature increases the F1 score of at least 19.89 percentage points. We hypothesise this may be an artefact of the way the dataset is constructed. With the inclusion of this feature, in our experimental setup, we find results of 62.10% F1 score, in cross-validation evaluation.

This paper is organised as follows. In Section 2 we present the background for our approach. In Section 3 we describe the data in the competition and the techniques we used for solving the problem. In Section 4 we explain our experimental setup and show our results. In Section 5 we discuss previous approaches in the literature and, finally, in Section 6 we present final considerations and discuss future work.

2 BACKGROUND

The usage of vector representations of text is traditional in the information retrieval literature, allowing the use of convenient linear-algebraic methods [Manning et al. 2008, pp. 107-123]. We use here two different kinds of vector representations of documents: tf-idf and dense sentence embeddings.

A traditional way to convert documents into vector representations is based on word counts, resulting in models called bag-of-words, as a document is a function only of which and how many words occur in it (but not of word order or any kind of organization in the text) [Manning et al. 2008, p. 107]. The simplest way of doing so is building a vector space in which every coordinate represents the number of times a specific word occurs in a text. As very common words are usually not considered informative with respect to distinguishing documents, a usual improvement is making the value correspondent to a word inversely proportional to the number of documents in the corpus in which the word appears [Manning et al. 2008, pp. 107-109]. Typically this is done by multiplying each coordinate i corresponding to a word w by the inverse document frequency (idf), defined by $idf_i = log \frac{N}{df_i}$, where N is the number of documents and df_i is the number of documents in which w appears.

A distinct way of calculating vector representations of texts is by using (dense) sentence embeddings. While tf-idf are long and sparse, there are other methods of calculating vector representations of text which produce short and dense vectors. Many methods were originally designed for vector representation of words [Bengio et al. 2003; Mikolov et al. 2013], based on the principle of trying to predict words' occurrence [Baroni et al. 2014] instead of counting co-occurrences of words. Inspired by these approaches, vector representations of sentences were developed as well [Cer et al. 2018a,b; Conneau et al. 2017; Hill et al. 2016; Kiros et al. 2015]. Initially, the literature on sentence embeddings was based only on manually unannotated text, using supervised learning in order, given a sentence, to predict the adjacent sentences [Conneau et al. 2017; Hill et al. 2016]. Building on these works, there are now approaches for including information derived from other linguistic tasks [Cer et al. 2018a,b; Conneau et al. 2017], especially natural language inference (NLI, also known as textual entailment) [Bowman et al. 2015]. The intuition is that sentence embeddings generated from supervised semantic tasks could capture this semantic information in the distributed representations. In particular, in this work we use the universal sentence encoder (USE) model, which is generated from multi-task learning, including a sentence prediction task, and semantic tasks, such as sentiment analysis and semantic textual similarity [Cer et al. 2018a,b]. That is, USE uses not only the non-manually annotated signal, but information from manually annotated tasks as well.

Given vector representations of two different texts, a typical method for comparing their similarity is by measuring the cosine between them [Manning et al. 2008, p. 111], which in vectors normalized by length corresponds simply to the dot product. Another measure is the length of the difference between the vectors [Conneau et al. 2017], although this is a simple transformation of the cosine if the vectors are normalized. Using either of those measures provides a single number which can be used as a (dis)similarity measurement in a way which is independent of the task at hand. However, given a training set for a specific task, we can use these similarity measures as features of a supervised machine learning task [Manning et al. 2008, pp. 315-320], such as nearest neighbours classification [Mitchell 1997, p. 231], random forests [Breiman 2001] and support vector machines [Cortes and Vapnik 1995]. This is what we do in this work.

3 DATA AND METHODS

The dataset for Task 1 of the COLIEE competition is drawn from Canadian Courts, mainly from the Federal Court of Canada [Kim et al. 2018, p. 2]. The task is presented as follows: there are a number of problem instances, where each instance consists of a legal decision (hereinafter "new case") in which every citation or reference to a past case was suppressed. Each instance also contains a set of 200 other legal decisions (from now on, "candidate cases"). Some of those were originally referenced in the new case ("noticed cases") and others were not. The goal is, for each instance, finding which decisions among the candidate cases are noticed cases. More formally, a solution should return, for each instance, a subset of its candidate cases. Then the returned solution is evaluated against the correct one by the F1 metric, the harmonic mean between precision and recall.

According to the description of the previous edition of the competition, the set of candidate cases is generated as follows: first the noticed cases are included, and then cases are (likely uniformly) randomly sampled from a database of cases[2] [Kim et al. 2018].

A training dataset is made available, containing in total 285 instances, with the extra information of the list of noticed cases[3]. There are a total of 10346 unique candidate cases in the training set, as the same decision may appear as a candidate case in different instances. The task itself is evaluated then on a test dataset, containing 61 instances, but now without the set of noticed cases (i.e. the answers). In the test set, there are a total 7109 unique candidate cases. Including every case (new cases and candidate cases) in the training set, the average number or words (tokens) is 2537.60, with standard deviation of 1689.69 words. For the test set, the average number of words is 2411.39, with standard deviation of 1609.18 words[4].

Our approach for this task consists in representing case texts as vectors and using such vectors to create features for supervised learning tasks. We transform this problem in such way to make it amenable to two-class classifier techniques. In particular, we make every pair (*new case*, *candidate case*) an element to be classified as either relevant or irrelevant (an approach presented by, e.g., Manning et al. [2008]). As the evaluation is not based on ranked predictions, only on binary ones, it is not necessary to consider the scoring between the presented noticed cases.

Noticing the repetition of candidate cases, we also explored with one extra feature: the number of instances (in the entire dataset) a particular candidate case appeared as a candidate. Due to the way the instances were created, we conjectured this value might present some bias.

More specifically, we consider two main scenarios: (A) random forests, using as features the cosine score between the vector representations of the new case and the candidate case; (B) k-nearest neighbours, using as features the entire vectors of the new case and of the candidate case. As the k-nearest neighbours method by itself considers the distance between the elements, we use the entire vectors as features in order to make it comparable to random forests using features for vector similarity directly. In each of those two scenarios we also consider our extra count feature.

This approach has some weaknesses for the task at hand. Ideally, we would like a classification system which considers hypotheses on the set of all noticed cases, that is, capturing the entire task for an instance at the same time. The formulation as a binary classification problem does not consider this, as using this approach the classification decides on each candidate case separately. This means that some aspects are limited. For instance, one example of information that could not be adequately used in this scenario is the number of noticed cases. Even if the exact number of noticed cases in a particular new case was known, this setup would not be able to guarantee that this exact number would be predicted as noticed cases, as each decision for a pair (*new case*, *candidate case*) is made without information on how the decision was made for the other candidate cases. Using a more precise method for this is left for future work.

4 EXPERIMENTS

We used the Python package Gensim to generate tf-idf vectors from all case texts from the entire dataset [Řehůřek and Sojka 2010], as well as the universal sentence encoder (USE) [Cer et al. 2018a,b], version 2, trained on the deep averaging network encoder. For the classification methods, we used random forests and k-nearest neighbours, implemented in the Python package Scikit-learn[5] [Pedregosa et al. 2011].

We explored the following hyperparameters: a) in random forests, the number of estimators (from 5 to 200) and the split criteria (information gain vs Gini impurity); b) in k-nearest neighbours, only the number of neighbours (from 5 to 20). Although we did not use over-sampling or under-sampling, we adjusted weights in the random forests in order to be inversely proportional to class frequency in the training data[6]. More specifically, in a dataset of size D, with m different classes, each one with D_c instances for class c, the weight of an instance of class c is defined as $\frac{D}{m*D_c}$.

We run our experiments in 5-fold nested cross-validation. In this procedure, 5-fold cross-validation is made for partitioning between a training set and a test set, and then the training set is partitioned (using 5-fold cross-validation as well) into properly training set and a validation set, for hyperparameter search. The best choice of hyperparameters is then trained on the entire (training and validation) training set, and evaluated on the test set. Following Forman and Scholz [2010], we report precision, recall and F1 scores aggregated from the cross-validation by summing the number of true positives, false positives and false negatives of every fold, and then using the definitions of the metrics, and not by averaging the F1 (or precision and recall) scores from each fold, as the average is shown to be biased. This is also the reason why we do not report standard deviation of the aggregated metrics.

The results are in Table 1. We then evaluate the top 3 models on the test set, and we report results on Table 2. The model we submitted (`CLARg`) was k-nearest neighbours using TF-IDF, USE and counts. However, the results here differ from our official submission (with 46.01% F1 score), as unfortunately there was a flaw in our system and thus our method had been incorrectly trained. We also report baseline results from retrieval based simply on the top-k most similar according to cosine distance, for both tf-idf vectors and for USE embeddings, in Table 3. As the method is unsupervised, we simply report results on the entire training dataset.

We find that the number of occurrences feature is responsible for much of the results, up to 19.89 percent points in random forests, and up to 29.22 percent points in k-nearest neighbours, when using both tf-idf and USE. The most dramatic difference is on tf-idf with k-nearest neighbours, 30.21%. We hypothesize that this is an artefact of the way the dataset is generated. One possibility is, for instance,

[2]The universe of cases included in this database is unclear.

[3]Notice that even though there is the list of noticed cases, it is not presented in which parts of the new case they were referred. That is, the correspondence between noticed cases and the new case text, originally occurring, is still absent. This creates an extra challenge from approaches based on, for instance, trying to guess the noticed cases by associating the candidate cases with the parts in which a reference was suppressed.

[4]The number of words is counted in tokens (not unique word types), and was calculated by using the tokenizer distributed with the Gensim package for Python.

[5]We experimented briefly with support vector machines as well, but without under-sampling negative examples their training time was impractically high.

[6]This is an already implemented feature in Scikit-learn: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html#sklearn.ensemble.RandomForestClassifier

Table 1: Results on the training dataset, on nested 5-fold cross-validation, for the supervised learning approaches.

Method (with features)	Precision	Recall	F1
Random forests - TF-IDF, USE, counts	78.19%	49.93%	60.94%
Random forests - USE, counts	51.36%	39.37%	44.57%
Random forests - TF-IDF, counts	59.37%	50.54%	54.60%
Random forests - TF-IDF, USE	57.37%	31.97%	41.05%
Random forests - USE	12.37%	11.91%	12.14%
Random forests - TF-IDF	33.18%	29.95%	31.48%
K-Nearest neighbours - TF-IDF, USE, counts	82.30%	46.64%	59.54%
K-Nearest neighbours - USE, counts	84.57%	49.06%	62.10%
K-Nearest neighbours - TF-IDF, counts	87.15%	35.60%	50.55%
K-Nearest neighbours - TF-IDF, USE	65.59%	20.52%	31.27%
K-Nearest neighbours - USE	65.01%	22.01%	32.88%
K-Nearest neighbours - TF-IDF	29.63%	00.54%	01.06%

Table 2: Results on the test dataset, for the supervised learning approaches.

Method (with features)	Precision	Recall	F1
Random forests - TF-IDF, USE, counts	78.64%	49.09%	60.45%
K-Nearest neighbours - TF-IDF, USE, counts	84.49%	47.88%	61.12%
K-Nearest neighbours - USE, counts	86.34%	47.88%	61.60%

Table 3: Results on the training dataset for the unsupervised top-k retrieval based on cosine similarity.

Vector representation	Number of retrieved (k)	Precision	Recall	F1
USE	5	28.21%	27.05%	27.62%
USE	10	20.39%	39.10%	26.80%
USE	15	16.51%	47.51%	24.51%
TF-IDF	5	**46.74%**	44.82%	**45.76%**
TF-IDF	10	32.21%	61.78%	42.34%
TF-IDF	15	25.22%	**72.54%**	37.42%

that there is some kind of bias in the sampling of "extra", non-noticed cases in order to fill the 200 candidates for every instance, making the non-noticed cases occur as candidates with different probability than noticed cases.

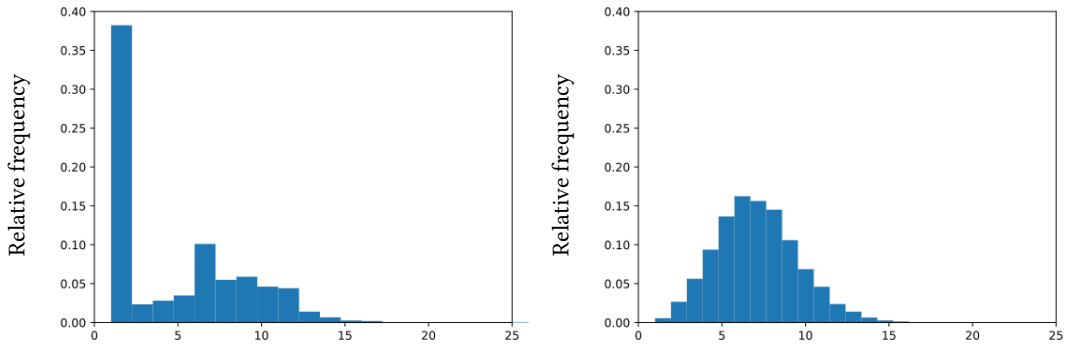
In order to investigate this distinction, we compare in Figure 1 the distribution of the number of occurrences feature between cases which are noticed in at least one instance (1195 cases), and cases which are never noticed (9151 cases). The graphs show the difference between them, and how having a low value (more specifically, value 1) in this feature can be suggestive that the case is a noticed one. This also suggests that there is some bias in sampling the candidate cases for making each entry, in that some noticed cases may not be part of the sampled universe of cases.

Lastly, in the test set, the models have particularly high precision (78% or more). This might be due to the number of occurrences feature. It could be argued that, for some natural use cases of a retrieval task, recall should have greater importance. While this is reasonable, the competition uses F1 as the final metric, as opposed to F_β scores with β greater than 1, which gives higher weight to recall.

5 RELATED WORK

According to the reports of the COLIEE 2018 edition, the winning method used document embeddings, generated from training a machine learning model on a summarization task, in which the summary (occurring in the original full text) of each case is considered a gold label, and the input is the text decision [Kim et al. 2018, p. 8]. The top-10 most similar candidates are retrieved. The second best method is by the same authors, using a relative number from the score deviation instead of strictly the top 10. The third best method was a random forest classifier on features based on doc2vec cosine similarity and tf-idf Euclidean distance. Other approaches used were association rules, classifier ensembles, over-sampling and under-sampling, and tf-idf selecting the k nearest candidates and applying a threshold on similarity in order to filter after top-k retrieval.

One important difference from the previous competition is that the new cases are now the complete text of a legal decision, with citations to past cases omitted. Previously, only the summary and the factual description part of the decision from the new cases were



(a) Noticed candidate cases (total of 1195 cases) **(b) Not noticed candidate cases (total of 9151 cases)**

Figure 1: Distribution of the number of occurrences feature, for two different groups: candidate cases which are noticed in at least one instance 1a, and candidate cases which are never noticed 1b.

available. For this reason, results from the previous competition are not directly comparable to results on the current one.

Regarding legal information retrieval in general, van Opijnen and Santos [2017] and Turtle [1995] discuss particular differences between legal retrieval and information retrieval in other domains, such as document size, document structure, the specificities of language (not only in vocabulary, but also in writing style), and the existence of a system as the context in which legal documents are generated, among others. van Opijnen and Santos [2017] discuss different dimensions of relevance for legal information retrieval, in the context of real systems.

Turtle [1995] presents an overview of information retrieval contextualized to the legal domain. A more recent presentation is in the work of Ashley [2017, chapter 7]. An example of a legal document management system, using cosine similarity on tf-idf vectors, is the work of Boella et al. [2016]. Koniaris et al. [2016] present an experiment on diversification heuristics in legal information retrieval, in order to provide non-redundant results.

Recent commercial legal information retrieval systems use features such as metrics from citation networks, expert-generated annotations, and user feedback, as well as lexical and conceptual information [Ashley 2017, chapter 7]. Other approaches include the usage of legal ontologies [Saravanan et al. 2009], as well as integration with legal case-based systems [Daniels and Rissland 1997a,b], such as Hypo and CATO [Aleven 2003; Ashley 1991].

6 CONCLUSION

In this work, we used vector representations of legal sentences for the task of information retrieval in case law, in the COLIEE competition. We explore Universal Sentence Encoder, a sentence embedding method, and the more traditional tf-idf vectors. We used features derived from vector representations to pose the problem as a binary classification task, which we solved with two different classifiers: random forests and k-nearest neighbours. We concluded that pre-trained sentence embedding features used in this way do provide improvements over using tf-idf only. A more sophisticated way of using sentence embeddings is by fine-tuning (or perhaps even retraining from scratch) the representation on legal text, which we leave for future work.

We also explored the impact of adding a feature consisting of how many times a particular past case has appeared in the entire dataset, noticing that it has a dramatic effect on the F1 score. Our findings with respect to the new feature may be useful for designing future competitions.

Possible future work is using neural networks as classifiers, which fits naturally with the use of sentence embeddings: in that way, sentence embeddings can be used as well not as fixed features, but as starting parameters in a neural network architecture, integrating embedding generation with solving the specific retrieval task. Another possible development is using more specific parts of the text, instead of treating each decision as a single input to be transformed into a vector. As the decisions have some sort of structure, being sensitive to it could improve results. Finally, while sentence embeddings may prove useful, they are not interpretable. The development of sentence embeddings which are in some sense human understandable would make them more appealing, following work which already started on word embeddings.

ACKNOWLEDGMENTS

The first author was supported by Capes (Brazil, Ph.D. Scholarship 88881.174481/2018-01).

REFERENCES

Vincent Aleven. 2003. Using background knowledge in case-based legal reasoning: A computational model and an intelligent learning environment. *Artificial Intelligence* 150, 1-2 (Nov 2003), 183–237. https://doi.org/10.1016/s0004-3702(03)00105-x

Kevin D. Ashley. 1991. Reasoning with cases and hypotheticals in HYPO. *International Journal of Man-Machine Studies* 34, 6 (Jun 1991), 753–796. https://doi.org/10.1016/0020-7373(91)90011-u

Kevin D. Ashley. 2017. *Artificial Intelligence and Legal Analytics.* Cambridge University Press. https://doi.org/10.1017/9781316761380

Marco Baroni, Georgiana Dinu, and Germán Kruszewski. 2014. Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors, See [The Association for Computer Linguistics 2014], 238–247. http://aclweb.org/anthology/P/P14/P14-1023.pdf

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. 2003. A Neural Probabilistic Language Model. *Journal of Machine Learning Research* 3 (2003), 1137–1155. http://jmlr.org/papers/v3/bengio03a.html

Yoshua Bengio and Yann LeCun (Eds.). 2013. *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*. https://openreview.net/group?id=ICLR.cc/2013

Eduardo Blanco and Wei Lu (Eds.). 2018. *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, EMNLP 2018: System Demonstrations, Brussels, Belgium, October 31 - November 4, 2018*. Association for Computational Linguistics. https://aclanthology.info/volumes/proceedings-of-the-2018-conference-on-empirical-methods-in-natural-language-processing-system-demonstrations

Guido Boella, Luigi Di Caro, Llio Humphreys, Livio Robaldo, Piercarlo Rossi, and Leon van der Torre. 2016. Eunomos, a legal document and knowledge management system for the Web to provide relevant, reliable and up-to-date information on the law. *Artif. Intell. Law* 24, 3 (2016), 245–283. https://doi.org/10.1007/s10506-016-9184-3

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing* (2015). https://doi.org/10.18653/v1/d15-1075

Leo Breiman. 2001. Random Forests. *Machine Learning* 45, 1 (01 Oct 2001), 5–32. https://doi.org/10.1023/A:1010933404324

Wojciech Cellary, Mohamed F. Mokbel, Jianmin Wang, Hua Wang, Rui Zhou, and Yanchun Zhang (Eds.). 2016. *Web Information Systems Engineering - WISE 2016 - 17th International Conference, Shanghai, China, November 8-10, 2016, Proceedings, Part I*. Lecture Notes in Computer Science, Vol. 10041. https://doi.org/10.1007/978-3-319-48740-3

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Brian Strope, and Ray Kurzweil. 2018a. Universal Sentence Encoder for English, See [Blanco and Lu 2018], 169–174. https://aclanthology.info/papers/D18-2029/d18-2029

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. 2018b. Universal Sentence Encoder. *CoRR* abs/1803.11175 (2018). arXiv:1803.11175 http://arxiv.org/abs/1803.11175

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. 2017. Supervised Learning of Universal Sentence Representations from Natural Language Inference Data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Copenhagen, Denmark, 670–680. https://doi.org/10.18653/v1/D17-1070

Corinna Cortes, Neil D. Lawrence, Daniel D. Lee, Masashi Sugiyama, and Roman Garnett (Eds.). 2015. *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*. http://papers.nips.cc/book/advances-in-neural-information-processing-systems-28-2015

Corinna Cortes and Vladimir Vapnik. 1995. Support-vector networks. *Machine Learning* 20, 3 (01 Sep 1995), 273–297. https://doi.org/10.1007/BF00994018

Jody J. Daniels and Edwina L. Rissland. 1997a. Finding legally relevant passages in case opinions. *Proceedings of the sixth international conference on Artificial intelligence and law - ICAIL '97* (1997). https://doi.org/10.1145/261618.261627

Jody J. Daniels and Edwina L. Rissland. 1997b. What you saw is what you want: Using cases to seed information retrieval. *Lecture Notes in Computer Science* (1997), 325–336. https://doi.org/10.1007/3-540-63233-6_503

George Forman and Martin Scholz. 2010. Apples-to-apples in Cross-Validation Studies: Pitfalls in Classifier Performance Measurement. *ACM SIGKDD Explorations Newsletter* 12, 1 (Nov 2010), 49. https://doi.org/10.1145/1882471.1882479

Felix Hill, Kyunghyun Cho, and Anna Korhonen. 2016. Learning Distributed Representations of Sentences from Unlabelled Data, See [Knight et al. 2016], 1367–1377. http://aclweb.org/anthology/N/N16/N16-1162.pdf

Mi-Young Kim, Yao Lu, Juliano Rabelo, and Randy Goebel. 2018. COLIEE-2018: Evaluation of the Competition on Case Law Information Extraction and Entailment. Retrieved April 2, 2019 from https://web.archive.org/web/20190410225614/https://sites.ualberta.ca/~rabelo/COLIEE2019/COLIEE2018_CL_summary.pdf

Ryan Kiros, Yukun Zhu, Ruslan Salakhutdinov, Richard S. Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Skip-Thought Vectors, See [Cortes et al. 2015], 3294–3302. http://papers.nips.cc/paper/5950-skip-thought-vectors

Kevin Knight, Ani Nenkova, and Owen Rambow (Eds.). 2016. *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*. The Association for Computational Linguistics.

Marios Koniaris, Ioannis Anagnostopoulos, and Yannis Vassiliou. 2016. Multi-dimension Diversification in Legal Information Retrieval, See [Cellary et al. 2016], 174–189. https://doi.org/10.1007/978-3-319-48740-3_12

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to information retrieval*. Cambridge University Press.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space, See [Bengio and LeCun 2013]. http://arxiv.org/abs/1301.3781

Tom M. Mitchell. 1997. *Machine learning*. McGraw-Hill. http://www.worldcat.org/oclc/61321007

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.

Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. ELRA, Valletta, Malta, 45–50. http://is.muni.cz/publication/884893/en.

M. Saravanan, Balaraman Ravindran, and S. Raman. 2009. Improving legal information retrieval using an ontological framework. *Artif. Intell. Law* 17, 2 (2009), 101–124. https://doi.org/10.1007/s10506-009-9075-y

Frederick Schauer. 2009. *Thinking Like a Lawyer: A New Introduction to Legal Reasoning*. Harvard University Press, Cambridge, Massachusetts.

William Tetley. 2000. Mixed jurisdictions: common law vs civil law (codified and uncodified). *Lousiana Law Review* 60, 3 (2000).

The Association for Computer Linguistics 2014. *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, June 22-27, 2014, Baltimore, MD, USA, Volume 1: Long Papers*. The Association for Computer Linguistics. http://aclweb.org/anthology/P/P14/

Howard R. Turtle. 1995. Text Retrieval in the Legal World. *Artif. Intell. Law* 3, 1-2 (1995), 5–54. https://doi.org/10.1007/BF00877694

Marc van Opijnen and Cristiana Santos. 2017. On the concept of relevance in legal information retrieval. *Artificial Intelligence and Law* 25, 1 (01 Mar 2017), 65–87. https://doi.org/10.1007/s10506-017-9195-8

Statutory entailment using similarity features and decomposable attention models

John Hudzina
Centre for Cognitive Computing,
Thomson Reuters
Eagan, Minnesota, USA
john.hudzina@thomsonreuters.com

Thomas Vacek
Centre for Cognitive Computing,
Thomson Reuters
Eagan, Minnesota, USA
thomas.vacek@thomsonreuters.com

Kanika Madan
Centre for Cognitive Computing,
Thomson Reuters
Toronto, Canada
kanika.madan@thomsonreuters.com

Tonya Custis
Centre for Cognitive Computing,
Thomson Reuters
Eagan, Minnesota, USA
tonya.custis@thomsonreuters.com

Frank Schilder
Centre for Cognitive Computing,
Thomson Reuters
Eagan, Minnesota, USA
frank.schilder@thomsonreuters.com

ABSTRACT

This paper evaluates TR Statues' approach to handle entailment against a small domain-specific data set from COLIEE 2019 Task 4. While other domains have seen advancements in Natural Language Inference (NLI) tasks, legal entailment has been difficult particularly for supervised approaches. The examine two supervised approaches against the entailment of the Japanese Civil Code. The first approach leverages a Random Forest model trained with similarity features from a question answering system developed for an unrelated task. The second approach applies transfer learning against a decomposable attention model. Transfer learning demonstrated a modest improvement over past attempts with the same deep learning architecture.

CCS CONCEPTS

• **Computing methodologies** → **Natural language processing**; **Probabilistic reasoning**.

KEYWORDS

natural language processing, neural networks, entailment

ACM Reference Format:
John Hudzina, Thomas Vacek, Kanika Madan, Tonya Custis, and Frank Schilder. 2019. Statutory entailment using similarity features and decomposable attention models. In *Proceedings of COLIEE 2019 workshop: Competition on Legal Information Extraction/Entailment (COLIEE 2019).* ACM, New York, NY, USA, 5 pages.

1 INTRODUCTION

The Competition on Legal Information Extraction/Entailment (COLIEE) focuses on the entailment of legal text. This paper focuses on the entailment task of bar exam questions about the Japanese Civil Code. Given the relevant articles from the Japanese Civil Code, the system must determine if the provide bar exam question entails 'Y' or contradicts 'N' the statutes.

This work explores two different approaches to textual entailment of civil law. The first approach implements a traditional NLP pipeline and extracts language features to classify the premise and hypothesis pairs. The second approach implements a deep learning model with unsupervised pre-trained embeddings. The pre-training task can be analogized to the principles of statutory construction. While our model revisited the decomposition attention architecture used by JAIST [14], the pre-training step demonstrated a 4.9% improvement in absolute accuracy.

While this task is defined for Japanese statutes, it holds importance to US laws also. Statutory interpretation remains a major function of the US court systems. When parties cannot agree on the statute's meaning, a judge applies the rules of statutory construction to interpret a statute [4] [13]. These rules are contextual and give the judge a means to understand the legislature's intent.

The foremost rule of statutory construction is the plain-meaning rule (Sutherland §46:1). This rule dictates that a word's ordinary meaning applies unless that meaning yields an absurd result. The primacy of this rule suggests that statutory interpretation will benefit from transfer learning approaches to natural language inference. Although there are a large number of particularities (necessitating a large volume to study them [13]), a system that could determine statutory entailment just at the level of plain language would have considerable value, allowing counsel and court officials to check the statutory implications of a theory of law.

COLIEE Task 4 focuses on situations where the statute's text provide sufficient context. Inferences drawn from the statue's own text are known as intrinsic aids [13]. Intrinsic aids exclude situations where an external source provides the necessary context, like a dictionary [8] or case law. The intrinsic aids similar to task 4 are as follows:

- **Noscitur a sociis** translates to "know by its partners." The rule determines the meaning based on surrounding words in context. Sutherland §47:16
- **Eiusdem generis** translates to "of the same kind." If a list contains several specific descriptors and one generic descriptor, the generic descriptor should be limited to the same

COLIEE 2019, June 21, 2019, Montreal, Quebec

class. This rule is similar to "Noscitur a sociis" in that the proceeding words bound the context. Sutherland §47:17
- **Superfluous**: The rule dictates that every clause and word conveys the legislator's intent. In other words, any hypothesis must consider all the statutes words in order to be a valid entailment. Sutherland §46:6
- **Conflict**: The rule states that the statute's parts must not conflict. In essence, the statute should be read as a whole where the subsequent parts should entail the proceeding parts. Sutherland §46:5
- **Similarity or difference**: If the statute references the same word in different parts, then the word should convey the same meaning. Sutherland §46:6

2 ENTAILMENT APPROACHES

We examine two approaches:

(1) Similarity features. This is a baseline method consisting in a number of human-crafted features derived from an off-the-shelf NLP4J pipeline. These human-crafted features were designed for a question-answering task and repurposed for this task.
(2) Decomposable attention. This is a deep learning method which builds on a previous COLIEE entry.

In both cases, words in the premise and hypothesis are compared and then the scores are aggregated to determine if entailment holds.

2.1 Similarity features

The feature-based entailment system under evaluation here does not attempt to model sophisticated aspects of entailment such as negation, role assignment, or party assignment. Rather, it simply models the liminal question of whether the context and hypothesis are related. While this is expected to be inadequate to the task, other entailment approaches have also used similarity or word overlap features in non-deep learning models[12].

Instead of building similarity features from scratch, the baseline model re-purposes a feature pipeline intended for question and answer generation. The general idea behind these features is that concepts (important tokens or short sequences of them) from the question are more likely to occur in certain positions within high-quality answers as compared to the position of matching concepts within lower quality answers. Concepts present in both the question and a high quality answer will often occur:

- Near the beginning of the candidate answer
- Near the root node of the parse tree for the candidate answer
- Near each other within the candidate answer

Removed from the original question-answering task, these features are derived from the context and hypothesis sentences. They formalize intuitions about positional relationships between premise and hypothesis. The features are various aggregations (max, min, mean) and facets (all tokens, only some classes) of the following:

- **Offset features**: Measure distance to hypothesis words from start of the statute text, scanning left to right. Aggregations and facets include: average and minimum distance, counting only nouns or all matches, and offsets measured in tokens versus characters.
- **Tree distance features**: Measure length of the shortest path for each pair of hypothesis words found in the statute's text. Matches are based on word lemmas and exclude stop words.
- **Tree depth features**: Measure distance from root node to each hypothesis word found in the statute's parse tree. Matches are based on word lemmas and exclude stop words.
- **Symmetric difference features**: Measure symmetric difference using a sliding window over the relevant statutes (natural text, left to right), counting how many tokens are present in the symmetric difference for every window. Window size is determined by the hypothesis length.

The features generated were used in a regression against the entailment labels using standard cross-validation methods. Random Forest models were considered.

While this model incorporates similarity features, the baseline model won't fully handle paraphrasing because it leverages only token and lemma features. COLIEE's 2018 data set included 18 example that exhibit paraphrased passages and or verbs [18]. Methods based on word embeddings, such as [14] could be expected to handle paraphrasing better.

2.2 Decomposable attention

While JAIST [14] applied decomposable attention models to legal entailment in the 2017 competition, the submission scored slight above baseline. The authors attributed the poor results to the small training set provided for task 4. The small data set also caused issues for supervised models in the 2018 competition [18]. Lack of data is not a unique problem in the legal domain as standard data sets focus on readily available web or news content [15].

When small data sets limit supervised learning, new solutions incorporate cross-domain training data [15], pre-training via an unsupervised learning model [1] [2], or multi-task learning [7]. These approaches train generalized models and then later are fine-tuned to a specific task. We find that the BERT pre-training approach yields a 4.7% accuracy improvement for this task, compared to a previous submission that did not use BERT.

Given the advances with cross-domain data sets and pre-training, we revisited decomposable attention models. The model architecture was identical to JAIST's 2017 submission [14]. Our submission incorporates AllenNLP's decomposable attention model [3] which includes the same attend, compare, and aggregate components as JAIST.

Although the general architecture remained the same as the JAIST-NLP attention model, our model implements two significant differences: pre-trained language model and sentence selection. The pre-trained language model learned legal terminology by predicting missing words in the statute text and the statute's next sentence. The sentence selection reduces the hypothesis input length to avoid truncation via the model's text embeddings.

2.2.1 Pretraining. Our model implements BERT embeddings because BERT's pre-training aligns well with the statutory rules of construction. BERT's pre-training executes two tasks: a mask language model (MLM) and the next sentence prediction. The MLM task aligns with the construction rules because it predicts masked words by their surrounding context. In other words, the masked word is *noscitur a sociis* or known by its partners, Sutherland §46:6.

A judge must infer a word's meaning based on the context of the surrounding words in the statute [13].

The next-sentence prediction task aligns well with the construction rules because this task checks the consistency between sentences. The next-sentence prediction task predicts if a random sentence is next or comes from elsewhere in the corpus. If a word exists in both sentences, then each instance of the word should convey the same meaning. The test mimics the similarity or difference rule in Sutherland §46:5. The next-sentence test also ensures the sentences do not conflict, like the Sutherland §46:6.

2.2.2 Sentence selection. Although the BERT pre-training relates well to the statutory rules of construction, the sentence selection dealt more with the selected tool's limitation than legal canon. The Hugging Face embedder truncated the inputs after a maximum sequence length [16]. Rather than design a new deep learning architecture, our approach limited the input to the three most relevant sentences.

The sentence selection is as follows. The data preparation script splits the relevant statutes into sentences. The script transforms the question and the statutes sentences into a vector with scikit learn's TfidfVectorizer [10]. Each statute sentence x was then compared to the hypothesis sentence y:

$$k(x, y) = x^\top y \tag{1}$$

The script ranks the hypothesis sentences and then selects the top three as input for the decomposable attention model.

Unfortunately, this approach runs counter to the rules of construction because the model does not consider the entire statute when deciding if the statute entails the claim [13]. The decomposable attention model typically compares two sentences instead of full paragraphs. Given that d is the embedding dimension and l is the sentence length. If $l > d$, then attention networks are not computationally efficient compared to other architectures like RNNs [9].

3 EXPERIMENTS

This section describes the experiments conducted for the COLIEE's statute entailment task, task 4. We submitted two runs for evaluation, one each from the similarity feature and the decomposable attention models, TRSimFeat and TRAttn, respectively. Table 1 lists the results from both approaches in bold.

3.1 Decomposable attention with MultiNLI training set

Before training a model against COLIEE's training data, we evaluated a multi-source training set because we wanted to understand if the "plain meaning rule" actually applies to the Japanese civil code. This experiment utilized the MultiNLI [15] training and development sets to build the model. Then this experiment evaluated the model against the entire COLIEE data set from H18 to H29.

This experiment ran the decomposable attention model provided by AllenNLP [3]. The run used the BERT's large uncased embedding retrieved from the Hugging Face repository [16]. The BERT embeddings where originally trained with 24 layer, 16 heads, and 1024 hidden dimensions [2]. The model used AllenNLP's tokenizer which does not produce word-pieces. Table 2 lists the settings for the decomposable attention model.

Submission	Correct	Accruacy
BaseLine	51/98	0.5204
UA_Ex	67	0.6837
KIS_3module	61	0.6224
IITP	58	0.5918
KIS_dic	58	0.5918
UA_Go	58	0.5918
KIS_frame	57	0.5816
DBSE	56	0.5714
JNLP.t=98	56	0.5714
TRAttn	**55**	**0.5612**
TRSimFeat	**52**	**0.5306**
JNLP.t=85	51	0.5204
EVORA1	50	0.5102
JNLP.t=78	48	0.4898
EVORA3	47	0.4796
EVORA2	44	0.4490

Table 1: COLIEE 2019 Task 4 Results

Hyper Param	Attend	Compare	Aggregrate
Input Dim	1024	2048	400
Layers	2	2	2
Hidden Dim	200	200	200,3
Activations	relu	relu	relu,linear
Dropout	0.2	0.2	0.2,0.0

Table 2: MultiNLI generalized model's hyper-parameters

Before attempting to train the attention model, the experiment held out the entire COLIEE training set (H18 to H29) to examine if a generalized training set was sufficient for entailing statutory text. Figure 1 shows the MultiNLI model's results in a confusion matrix. If we assume the neutral class corresponds to a "No" answer, then the general NLI model's accuracy is 0.534, which is slightly above the 0.5 baseline.

While the model scored above the baseline with the holdout set, the experiment shows that a generalized training set is not sufficient for legal entailment. The model misclassified the hypothesis 17% of the time meaning the general model did not pick up paraphrasing and or role assignment. Then again, a neutral sentence is not necessarily wrong because there might not be enough context in the premise to support the hypotheses. In U.S. law, this gap would be filled by an external source like past case law or related statutes [13].

The general model also misclassified the "No" as "Yes" 65% of the time. The data set differences might explain these errors because the MultiNLI data set focuses on sentence pairs instead of full paragraphs. If multiple sentences in the article support the hypothesis, then those sentences might out weigh the sentence that contradicts the hypothesis. A data set with paragraph style premises, like the SQuAD 2.0 [11], might provide better results.

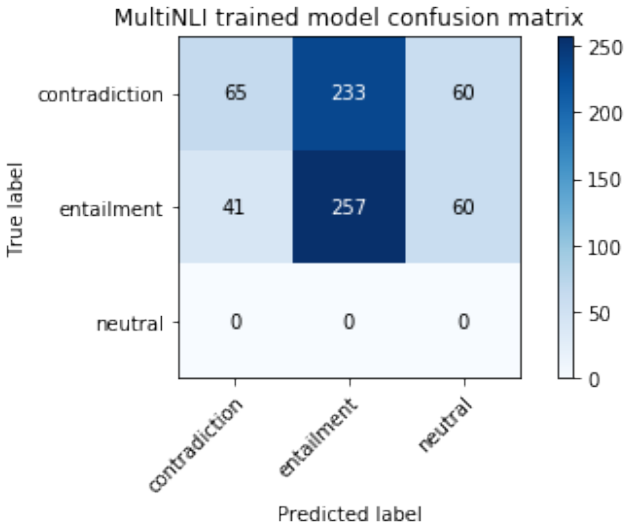


Figure 1: MultiNLI results for sentence pairs in H18 to H29

Hyper Param	Settings
Number of Trees	2,5,10,20
impurity	entropy,gini
Mix Bins	8,16,32,64

Table 3: Random forest hyper-parameter grid search

Ultimately, we did not submit this model's results for evaluation by COLIEE. In hindsight, we should have submitted the run because it would have set a benchmark for generalizability of the MultiNLI dataset.

3.2 Similarity features

Before attempting the decomposable attention model with the COLIEE training set, the next experiment established a baseline with a non-deep-learning model. This approach serves as a comparison because it models text similarity between the hypothesis and premises like decomposable attention model. The model incorporates word similarity and position features discussed in Section 2.1.

Before training the similarity model, we used the MultiNLI results to create a stratified validation set of 100 examples. The strata included the neutral misclassifications from Figure 1 which contained examples that represented paraphrasing. The same training and validation are used in the next experiment because we want to compare model.

This experiment trained the text similarity model with the following procedure. The feature extraction pipeline (Section 2.1) ran against the training and validation set. Using the training set, we trained a random forest model [5] using ten cross-fold validation. The training included the hyper-parameter options using the grid search parameters list in Table 3.

The similarity performed surprisingly well considering it only included token/lemma based similarity features. The model score just above the baseline with 52/98 answers correct, Table 1. The simple model did not include features that would be in a more robust model like negation, paraphrasing, and complement verb matching [12].

Hyper Param	Attend	Compare	Aggregrate
Input Dim	768	1536	400
Layers	2	2	2
Hidden Dim	200	200	200,2
Activations	relu	relu	relu,linear
Dropout	0.2	0.2	0.4,0.0

Table 4: COLIEE specific model's hyper-parameters

3.3 Decomposable attention with top-n sentences

While the similarity model preformed well given its simplistic token based features, the next experiment focused on paraphrasing via embedding based similarity. Unlike the MultiNLI experiment, the next decomposable attention experiment trained the model from exclusively the COLIEE provided data. We used the MultiNLI results to create a stratified validation set of 100 examples. The validation set included six strata which covered the categories with values in Figure 1. Two strata included examples that where missed classified as neutral because those misses contained examples with paraphrasing and or role assignment.

Before training the model, this experiment fined-tuned a language model using BERT's base uncased embeddings from the Hugging Face repository [16]. Google trained the BERT base uncased model with 12 layers, 12 heads, and 768 hidden dimensions[2]. We then ran Hugging Face's language model fine-tuning job initialize with the BERT parameters. The fine-tuning task reads in sentence pairs from the Japanese Civil Code and sentence pairs extracted from the training set. The fine-tuning ran 75 epochs with a learning rate of 3e-5 and a 128-word piece max sequence. The resulting model generates word-piece embeddings with 768 dimensions.

Once the pre-training was complete, the fine-tuned BERT embeddings where incorporated into AllenNLP's decomposable attention model[3]. The BERT embeddings required some minor modifications, including customizing the embeddings to use word-pieces instead of tokens. The word pieces are common sub-parts of words and allow for smaller vocabularies [17]. The embeddings used Hugging Faces' word-piece tokenizer directly so that the BERT vocabulary aligned with the tokens.

Before we trained the attention model, the premise/statutory text required some pre-processing because the word-piece tokenizer truncated long sequences. If the tokenizer truncated a critical sentence from the statute, then the model would misclassify the hypothesis. Instead, the statutes were split up into sentences and then ranked base on a TF-IDF similarity score to the hypothesis. The top three sentence where then feed into the model to avoid truncating a critical sequence.

After the sentence ranking, we trained the decomposable attention model with the component settings listed in Table4.

3.3.1 Comparison to the JAIST 2017 model. As noted previously, JAIST attempted the same model architecture for COLIEE 2017. Although the test sets differ between years, this section will compare both models to analyze any possible improvements for this approach. Compared to the JAISTNLP 2017 results[6], TRAttn (Table 1) showed some minor improvements from 0.512 (best run) to 0.561 accuracy.

Although TRAttn demonstrated a slight increase in accuracy, it is difficult to say that pretraining showed any improvements because the fundamental issues discussed by the JAIST team still exist. COLIEE's data sets are extremely small compared to other natural language inference data sets. Both SNLI and MultiNLI include 550K and 392K training examples, respectively [15]. In contrast, COLIEE provides 700 training examples for task 4. The small training set size leads to unstable supervised learning results[14][2] and increases the likelihood of over- or under-fitting.

4 DISCUSSION

The experiment results showed that supervised learning approaches remain difficult because the results remain unstable. While both approaches scored above the baseline, neither demonstrated state of the art performance. Given the relatively small size compare to other entailment training sets, improvements probably won't come for deep learning models built from the COLIEE training set alone. For a deep learning approach to work, a model will need to incorporate either transfer and or multi-task learning.

Although the TRAttn model show a minor improvement over JAISTNLP with pre-training, the pre-training wasn't sufficient to model every sub-task for entailment. The language model pre-training mimics some intrinsic aids, like *noscitur a sociis*, through predicting missing words and the next sentence. While the intrinsic aids help judges interpret statues, entailment includes other tasks like conditional, paraphrasing, case role assignment, person relationship, and person role assignment [18].

If statutory entailment requires multiple sub tasks, then multi-task learning provides a potential route for improvement. The BERT paper[2] describe other pre-training tasks, include SQuAD, which could be adapt to mimic the tasks required for full statutory entailment. These tasks need not be entirely based in the legal domain. If fact, the compliance with the "plain meaning" rule dictates some general knowledge to infer the statute's ordinary meaning.

5 SUMMARY

This paper describes our submissions from the COLIEE 2019's Japanese Civil Code entailment task. Given the relevant articles from the Japanese Civil Code, the system must predict if the provide bar exam question entails or contradicts the statutes. We submitted prediction from two separate models: similarity features and decomposable attention model with pre-trained embeddings. Although both models scored above the baseline, the decomposable attention demonstrated demonstrated a 4.9% improvement in absolute accuracy compare a previous years submission with the same architecture.

ACKNOWLEDGMENTS

To Kanika Madan, for the help with the question and answer pipeline.

REFERENCES

[1] Andrew M Dai and Quoc V Le. 2015. Semi-supervised Sequence Learning. In *Advances in Neural Information Processing Systems 28*, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett (Eds.). Curran Associates, Inc., 3079–3087. http://papers.nips.cc/paper/5949-semi-supervised-sequence-learning.pdf

[2] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv:1810.04805 [cs]* (Oct. 2018). http://arxiv.org/abs/1810.04805 arXiv: 1810.04805.

[3] Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson Liu, Matthew Peters, Michael Schmitz, and Luke Zettlemoyer. 2018. AllenNLP: A Deep Semantic Natural Language Processing Platform. *arXiv:1803.07640 [cs]* (March 2018). http://arxiv.org/abs/1803.07640 arXiv: 1803.07640.

[4] Maxine D. Goodman. 2004. Reconstructing the Plain Language Rule of Statutory Construction: How and Why. *Mont. L. Rev.* 65, 2 (2004).

[5] Tin Kam Ho. 1995. Random Decision Forests. In *Proceedings of the Third International Conference on Document Analysis and Recognition (Volume 1) - Volume 1 (ICDAR '95)*. IEEE Computer Society, Washington, DC, USA, 278–. http://dl.acm.org/citation.cfm?id=844379.844681

[6] Yoshinobu Kano, Mi-Young Kim, Randy Goebel, and Ken Satoh. 2017. Overview of COLIEE 2017. In *COLIEE 2017. 4th Competition on Legal Information Extraction and Entailment, held in conjunction with the 16th International Conference on Artificial Intelligence and Law (ICAIL 2017) in King's College London, UK*. 1–8. http://www.easychair.org/publications/paper/350845

[7] Bryan McCann, Nitish Shirish Keskar, Caiming Xiong, and Richard Socher. 2018. The Natural Language Decathlon: Multitask Learning as Question Answering. *arXiv:1806.08730 [cs, stat]* (June 2018). http://arxiv.org/abs/1806.08730 arXiv: 1806.08730.

[8] Sandra D. O'Conner. 1993. Smith v. U.S.

[9] Ankur Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. 2016. A Decomposable Attention Model for Natural Language Inference. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Austin, Texas, 2249–2255. https://aclweb.org/anthology/D16-1244

[10] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.

[11] Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know What You Don't Know: Unanswerable Questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Association for Computational Linguistics, Melbourne, Australia, 784–789. https://www.aclweb.org/anthology/P18-2124

[12] Nidhi Sharma, Richa Sharma, and Kanad K. Biswas. 2015. Recognizing Textual Entailment using Dependency Analysis and Machine Learning. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Student Research Workshop*. Association for Computational Linguistics, Denver, Colorado, 147–153. https://doi.org/10.3115/v1/N15-2020

[13] Norman J. Singer and Shambie Singer. 2014. *Sutherland Statutes and Statutory Construction* (7th ed.). Vol. 2A. Thomson Reuters.

[14] Nguyen Truong Son, Viet-Anh Phan, and Nguyen Le Minh. 2017. Recognizing entailments in legal texts using sentence encoding-based and decomposable attention models. In *COLIEE 2017. 4th Competition on Legal Information Extraction and Entailment, held in conjunction with the 16th International Conference on Artificial Intelligence and Law (ICAIL 2017) in King's College London, UK*. 31–42. http://www.easychair.org/publications/paper/347231

[15] Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference. In *Proceedings of NAACL-HLT 2018*. Association for Computational Linguistics, New Orleans, Louisiana, 1112–1122. https://doi.org/10.18653/v1/N18-1101

[16] Thomas Wolf. 2018. PyTorch Pretrained BERT: The Big & Extending Repository of pretrained Transformers. https://github.com/huggingface/pytorch-pretrained-BERT

[17] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, ÅĄukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. *arXiv:1609.08144 [cs]* (Sept. 2016). http://arxiv.org/abs/1609.08144 arXiv: 1609.08144.

[18] Masaharu Yoshioka, Yoshinobu Kano, Naoki Kiyota, and Ken Satoh. 2018. Overview of Japanese Statute Law Retrieval and Entailment Task at COLIEE-2018. https://sites.ualberta.ca/~rabelo/COLIEE2019/COLIEE2018_SL_summary.pdf

Searching Relevant Articles for Legal Bar Exam by Doc2Vec and T F-IDF

Ryuji Hayashi
Shizuoka University
Shizuoka, Japan
rhayashi@kanolab.net

Yoshinobu Kano
Shizuoka University
Shizuoka, Japan
kano@inf.shizuoka.ac.jp

ABSTRACT

We implemented an information retrieval system for legal domain using Doc2Vec to calculate document similarities, and TF-IDF to select keywords. We tried variations of our system: output the first-ranked similarity article only, output articles of similarities more than a fixed threshold, and output articles of similarities more than a dynamic threshold calculated from the top-ranked similarity score. These thresholds and Doc2Vec parameters are determined by a grid search with the training data of H18 to H29. We applied our system to COLIEE 2019 Task 3, which asks to find related Civil Code articles to solve a given Japanese legal bar exam problem. The result of our formal run was 0.4036 in the f2 score. We confirmed that the dynamic threshold is effective. Furthermore, we confirmed that the accuracy is improved by setting the vocabulary restriction by the value of TF-IDF in each document.

CCS CONCEPTS

• Information systems → Document structure; Similarity measures.

KEYWORDS

Doc2Vec, TF-IDF, COLIEE, Legal NLP

ACM Reference format:
Ryuji Hayashi and Yoshinobu Kano. 2019. Searching Relevant Articles for Legal Bar Exam by Doc2Vec and TF-IDF, In Proceedings of COLIEE 2019 workshop, ICAIL 2019, Montreal, Canada

1 Inroduction

The COLIEE (Competition for Legal Information Extraction and Entailment) workshop series has been held since 2014 [1][2][3][4]. Among four tasks in this COLIEE 2019 workshop, we focus on Task 3, which is an information extraction task using the Japanese Legal Bar Exam. Task 3 participants are required to output related articles of the Japanese civil code, in order to answer a given legal bar exam problem. The COLIEE organizers provide Japanese and English translation for problems and the civil code articles, while we use the Japanese version only.

COLIEE 2019, June 21, 2019, Montreal, Quebec

In order to obtain related article, we employ Doc2Vec [5], which can measure degree of similarity between documents. We also suggest a method to restrict vocabulary by TF-IDF[6] when training Doc2Vec.

There have been no approach that uses Doc2Vec in the past COLIEE participants. Team SPABS in COLIEE 2018 uses Word2Vec [7], which can calculate similarity between words. For training word embedding, they used legal documents with Word2Vec. SPABS_bm25 was their baseline results using BM25 [4]. SPABS used recurrent neural network to calculate similarity between questions and articles. Our research differs from this previous research in that we use doc2vec instead of word2vec, and uses TF-IDF instead of BM25 as a method to measure the importance of words.

Among various configurations we tried, the best method answers articles whose similarity score difference from the maximum similarity score is more than a threshold value. In addition, we obtained a better score when restricting vocabulary by TF-IDF for training Doc2Vec. The result of our formal run was 0.4036 in the f2 score.

We describe our system in Chapter 2, the experimental results in Chapter 3, analysis and discussion in Chapter 4, concluding the paper in Chapter 5.

2 System

We submitted two versions, KIS and KIS_2, and first we will explain the KIS_2 system.We used Doc2Vec and TF-IDF as a core of our system. We used morphological analyzer Juman[8] to segment civil code texts and past problem sentences. Then we select important words according to the value of TF-IDF. The selector in TF-IDF discards words whose values are less than a certain threshold. The same morphological analysis and word selection are performed on the test data, calculating the degree of similarity with the Civil Code article texts using the model trained by the training data and the Civil Code articles. In similarity calculation, cosine-similarity is calculated by doc2vec. We regard each article, t1, t2 texts as a document to be processed by Doc2Vec. In doc2vec, embedding was done for each article of the Civil Code text and for each question text as a single document.We select words whose values are less than a threshold when compared with the maximum similarity score. Figure 1 shows a conceptual flow of our system.

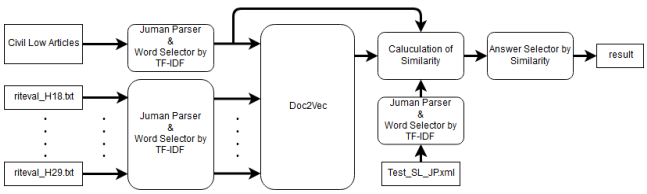


Figure 1 Steps to extract related articles

The KIS version system does not apply TF-IDF vocabulary restrictions to the KIS_2 system.

3 Experiments

We tried several different settings as follows. Firstly, we implemented our system only with Doc2Vec without restricting lexicon. By using this system, we tried to answer an article with top similarity, answer articles whose similarity values are more than a threshold value (0.60), and answer articles whose similarity value differences from the top article are less than a certain threshold value (0.065). The two threshold values of 0.6 and 0.065 are obtained by a grid search with intervals of 0.05 and 0.005, respectively, searching for a value that gives the best average score in the training data from H18 to H29. Table 1 shows the results of the COLIEE 2019 Task 3 formal run.

Table 1 Results of COLIEE 2019 Task 3formal run

team	f2
DBSE	0.4116
EVORA1	0.4811
EVORA2	0.4811
EVORA3	0.4811
HUKB	0.378
iitpBM25	0.4124
iitptfidf	0.3694
JNLP-tf	0.4931
JNLP-tfnv	0.4656
KIS	0.3885
KIS_2	0.4036
UA_LM	0.4124
UA_TFIDF	0.4983

Table 2 shows a set of parameters we used in Doc2Vec. The parameters in Table 2 were tuned for the average f2 value using the training data from H18 to H29. We increased the vector size by 200 to 25 and the epoch by 100 to 25 and looked for appropriate values.

Table 2 Doc2Vec parameter settings

Parameter	Value
vector_size	300
workers	50
window	50
alpha	0.025
min_alpha	0.0001
min_count	1
epochs	200

4 Analysis and Discussion

Tables 3, 4, and 5 show the evaluation results of past problems in the training data. For each year, we trained the system by the Civil Code articles and past questions other than the corresponding year which is used for evaluation.

Table 3 Evaluation resuls in training data by Doc2Vec (output Top 1 article)

Year	f2
H18	0.2388
H19	0.4116
H20	0.4471
H21	0.5765
H22	0.6298
H23	0.4764
H24	0.4735
H25	0.6030
H26	0.4929
H27	0.4155
H28	0.4392
H29	0.5348
average	0.4782

Table 3 Evaluation resuls in training data by Doc2Vec (fixed threshold)

Year	f2
H18	0.2938
H19	0.4500
H20	0.5295
H21	0.6058
H22	0.6040
H23	0.5195
H24	0.4804
H25	0.6340
H26	0.5439
H27	0.4126
H28	0.5476
H29	0.6586
average	0.5233

Table 4 Evaluation resuls in training data by Doc2Vec (dynamic threshold value)

Year	f2
H18	0.4115
H19	0.5596
H20	0.5741
H21	0.7144
H22	0.6879
H23	0.5945
H24	0.5745
H25	0.7182
H26	0.5810
H27	0.5537
H28	0.5918
H29	0.7120
average	0.6061

Firstly, H18 did not give good results throughout Tables 3 to 5. These problems would have had a different tendency as these are relatively old data. As shown in Table 2, the method of outputting only TOP 1 article results in the f2 value of at most 0.6298, which is not a good result. As shown in Table 3, the method allowing multiple answers with a fixed threshold value shows better f2 values in their average. However, the highest f2 value was 0.6586, and the maximum f2 value did not increase so much. Next, as shown in Table 4, our method with dynamic threshold value based on the Top similarity value, shows f2 values exceeding 0.71 in three years; all the years except H18 are f2 values of 0.55 or more. Secondly, we tried limiting vocabularies by TF-IDF, for both of the training documents (past problems and civil code articles) and the test data documents. When a TF-IDF value is 0.03 or less, the corresponding word is discarded from the keyword set. This threshold value, 0.03, for limiting vocabulary with TF-IDF was also obtained by searching for a value that results in the best average results tested from H18 to H29 with interval of 0.005. The results are shown in Tables 6, 7, and 8.

Table 5 Doc2Vec with TF-IDF(output TOP1)

Year	f2
H18	0.3663
H19	0.4422
H20	0.3787
H21	0.5943
H22	0.6723
H23	0.4764
H24	0.4556
H25	0.6197
H26	0.4823
H27	0.4592
H28	0.4724
H29	0.5638
average	0.4986

Table 6 Doc2Vec with TF-IDF(Fixed reference value)

Year	f2
H18	0.3889
H19	0.5055
H20	0.5034
H21	0.6120
H22	0.6667
H23	0.4984
H24	0.4691
H25	0.6398
H26	0.5422
H27	0.4700
H28	0.5434
H29	0.6317
average	0.5392

Table 7 Doc2Vec with TF-IDF(changed reference value)

Year	f2
H18	0.4877
H19	0.5675
H20	0.5889
H21	0.6840
H22	0.7137
H23	0.6278
H24	0.5696
H25	0.7020
H26	0.5968
H27	0.5613
H28	0.6203
H29	0.7233
average	0.6202

In all methods, our TF-IDF vocabulary restriction increased the average f2 values. However, scores were not always better in all individual years. In Table 8, for example, H21, H24 and H25 showed lower scores than the results of Table 5.

We observed characteristics of correctly answered problems and wrongly answered problems for H29. As a matter of course, we

were able to get perfect answers for questions that frequently use the same word as the problem text, and questions that do not include many words that are not used in the problem text. In addition, if there are many sentences or words that are not included in the problem text, e.g. when a concrete example is described, or if the problem text is considerably shorter than the gold standard related texts, or if there are texts that are quite similar, our system could not answer correctly. For many other problems, one of the multiple system outputs was correct, resulting in a partial correct answer.

Here are some example problems which our system could not answer. Figure 2 is an example which our system could not correctly answer, where a problem specific expression appears that is not described in the problem text.

Japanese text
<t1>
第九十四条
相手方と通じてした虚偽の意思表示は、無効とする。
２
前項の規定による意思表示の無効は、善意の第三者に対抗することができない。
</t1>
<t2>
甲土地を所有するＡがＢと通謀して甲土地をＢに仮装譲渡し，ＡからＢへの所有権移転登記がされた後，Ｂの債権者Ｃが甲土地を差し押さえた場合において，その差押えの時にＣが仮装譲渡について善意であったときは，Ａは，Ｃに対し，Ｂへの譲渡が無効であることを主張することができない。
</t2>

English text
<t1>
Article 94
(1) Any fictitious manifestation of intention made in collusion with another party(ies) shall be void.
(2) The nullity of the manifestation of intention pursuant to the provision of the preceding paragraph may not be asserted against a third party without knowledge.
</t1>
<t2>
In the case where, after A, who owns the land "X", had faked a transfer of the land "X" to B based on collusion with B and the transfer of ownership from A to B had been registered, B's obligee C attached the land "X", if C was without knowledge with respect to the fake transfer at the time of the attachment, A may not assert, against C, that the transfer to B is void.
</t2>

Figure 2 Example problem which our system wrongly answered (H29-4-A)

Figure 3 is an example in which the problem sentences were written fairly simply, so our system could not answer correctly.

Japanese text
<t1>
第六百三十九条
第六百三十七条及び前条第一項の期間は、第百六十七条の規定による消滅時効の期間内に限り、契約で伸長することができる。
（請負人の担保責任の存続期間）
第六百三十七条
前三条の規定による瑕疵の修補又は損害賠償の請求及び契約の解除は、仕事の目的物を引き渡した時から一年以内にしなければならない。
２
仕事の目的物の引渡しを要しない場合には、前項の期間は、仕事が終了した時から起算する。
第六百三十八条
建物その他の土地の工作物の請負人は、その工作物又は地盤の瑕疵について、引渡しの後五年間その担保の責任を負う。ただし、この期間は、石造、土造、れんが造、コンクリート造、金属造その他これらに類する構造の工作物については、十年とする。
２
工作物が前項の瑕疵によって滅失し、又は損傷したときは、注文者は、その滅失又は損傷の時から一年以内に、第六百三十四条の規定による権利を行使しなければならない。
</t1>
<t2>
請負人の担保責任の存続期間は，これを契約で伸長することができない。
</t2>

English text
<t1>
Article 639
The periods set forth in Article 637 and Paragraph 1 of the preceding Article may be extended by contract so long as they do not exceed the period of time provided for the extinctive prescription under the provisions of Article 167.
(Duration of Contractor's Warranty)
Article 637
(1) The demand for repair or claim for damages and cancellation of the contract under the preceding three Articles must be made within one year from the time of the delivery of the subject matter of the work.
(2) Where no delivery of the subject matter is required, the period referred to in the preceding paragraph commences to run from the time of the completion of the work.
Article 638
(1) A contractor for a building or other structure on land shall be liable for a warranty against defects in the structure

or ground for the period of five years from delivery; provided, however, that the period shall be ten years for structures made of stone, earth, bricks, concrete, steel and other similar structures.
(2) If any structure is lost or damaged due to the defects set forth in the preceding paragraph, the party ordering the work must exercise the rights under the provisions of Article 634 within one year from the time of the loss or damage.
</t1>
<t2>
The duration of contractor's warranty may not be extended by contract.
</t2>

Figure 3 Example problem which our system wrongly answered (H29-28-E)

Finally, Figure 4 is an example in which our system derived another sentence, and could not answer correctly because there was a very similar article. The correct answer is 666, but the system output 662.

Japanese text
<t1>
（消費寄託）
第六百六十六条　第五節（消費貸借）の規定は、受寄者が契約により寄託物を消費することができる場合について準用する。
２　前項において準用する第五百九十一条第一項の規定にかかわらず、前項の契約に返還の時期を定めなかったときは、寄託者は、いつでも返還を請求することができる。
</t1>
<t2>
消費寄託における寄託者は，寄託物の返還時期の定めがあるときであっても，いつでも寄託物の返還を請求することができる。
</t2>

English text
<t1>
(Deposits for Consumption)
Article 666
(1) The provisions of Section 5 (Loans for Consumption) shall apply mutatis mutandis to cases where a depositary may, under the contract, consume the Thing deposited.
(2) Notwithstanding the provisions of Paragraph 1 of Article 591 which shall apply mutatis mutandis under the preceding paragraph, if the contract referred to in the preceding paragraph does not specify the timing of the return, the depositor may demand the return at any time.
</t1>
<t2>
Even if the timing of the return of the Thing deposited is specified, the depositor in cases of a deposit for consumption may demand the return of the Thing deposited at any time.
</t2>

Japanese text
（寄託者による返還請求）
第六百六十二条　当事者が寄託物の返還の時期を定めたときであっても、寄託者は、いつでもその返還を請求することができる。

English text
(Depositor's Demand for Return)
Article 662 Even if the parties specify the time for the return of the Thing
deposited, the depositor may demand the return of the same at any time.
(Timing of Return of the Thing Deposited)

Figure 4 Example problem which our system wrongly answered(H29-29-O)

Regarding the case of Figure 4, our system output Article 662 because the text inside parenthesis written before the Article No. was not included in the training data. Articles are quite similar from Article 662 to Article 666, but the contents in parentheses are important in characterizing each article. In the case of Figure 4, the important words cannot be recognized and output 662, because “Deposits for Consumption” is not included in the training data, while the other sentences are similar.

Regarding the threshold values, the average accuracy increased when the dynamic threshold was used rather than fixed. We confirmed that the accuracy was improved by selecting the vocabulary when training documents. Because training with the Civil Code articles only could have provided a certain degree of good accuracy, Doc2Vec should be a suitable method to retrieve related texts.

4 Conclusion and Future Work

We implemented an information retrieval system for legal domain using Doc2Vec to calculate document similarities, and TF-IDF to select keywords. We tried variations of our system: output the first-ranked similarity article only, output articles of similarities more than a fixed threshold, and output articles of similarities more than a dynamic threshold calculated from the top-ranked similarity score. These thresholds and Doc2Vec parameters are determined by a grid search with the training data of H18 to H29.

We applied our system to COLIEE 2019 Task 3, which asks to find related Civil Code articles to solve a given Japanese legal bar exam problem. The result of our formal run was 0.4036 in the f2

score, which was 6th place among 7 participation teams.We confirmed that the dynamic threshold is effective. Furthermore, we confirmed that the accuracy is improved by setting the vocabulary restriction by the value of TF-IDF in each document.

We started our implementation several weeks before the submission deadline, so there are still many future works we wish to try. The results of any of our method vary from year to year. We would like to analyze the trend of the problems for each target year, and choose what method is effective to obtain a higher correct answer rate stably. We also plan to try using other documents for training Doc2Vec.

ACKNOWLEDGMENTS

This research was partially supported by JST CREST and MEXT kakenhi.

REFERENCES

[1] M.-Y. Kim, R. Goebel, and S. Ken, "COLIEE-2015: Evaluation of Legal Question Answering," in Ninth International Workshop on Juris-informatics (JURISIN 2015), 2015.

[2] M.-Y. Kim, R. Goebel, Y. Kano, and K. Satoh, "COLIEE-2016: Evaluation of the Competition on Legal Information Extraction/Entailment," in Tenth International Workshop on Juris-informatics (JURISIN 2016), 2016.

[3] Y. Kano, M.-Y. Kim, R. Goebel, and K. Satoh, "Overview of coliee 2017," Epic Ser. Comput., vol. 47, pp. 1–8, 2017.

[4] M. Yoshioka, Y. Kano, N. Kiyota, and K. Satoh, "'Overview of Japanese Statute Law Retrieval and Entailment Task at COLIEE-2018,'" in Twelfth International Workshop on Juris-informatics (JURISIN 2018), 2018.

[5]Le, Q. and Mikolov, T. , "Distributed Representations of Sentences and Documents", CoRR, abs/1405. 4053, pp. 1-9, 2014

[6]Salton, G., Wong, A., and Yang, C. S.: A Vector Space Model for Automatic Indexing.
Commun. ACM, Vol. 18, Num. 11, pp. 613-620 (1975)

[7] Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in neural
information processing systems. (2013)

[8] "JUMAN - KUROHASHI-KAWAHARA LAB" http://nlp.ist.i.kyoto-u.ac.jp/index.php?JUMAN (May 10, 2019)

Textual entailment using word embeddings and linguistic similarity

Kanika Madan
Centre for Cognitive Computing, Thomson Reuters
Toronto, Canada
kanika.madan@thomsonreuters.com

John Hudzina
Centre for Cognitive Computing, Thomson Reuters
Eagan, Minnesota, United States
john.hudzina@thomsonreuters.com

Thomas Vacek
Centre for Cognitive Computing, Thomson Reuters
Eagan, Minnesota, United States
thomas.vacek@thomsonreuters.com

Frank Schilder
Research Development, Thomson Reuters
Eagan, Minnesota, United States
frank.schilder@thomsonreuters.com

Tonya Custis
Research Development, Thomson Reuters
Eagan, Minnesota, United States
tonya.custis@thomsonreuters.com

ABSTRACT

This paper introduces our system submitted to the 2019 Competition on Legal Information Extraction and Entailment 2019 (COLIEE-2019), an associated event of ICAIL 2019. This year, COLIEE had four tasks, which focused on legal information processing and finding textual entailment on textual data from legal cases and statute laws. In this paper, we present our approach on the Task 2 of finding textual entailment relationships between the legal cases. We built a ranking algorithm that used a combination of word embeddings and other textual similarity features to find entailment relationships and to select the paragraph which is most probable to contain the entailment relationship. We also provide a comparison of our ranking approach with other methods that we tried.

CCS CONCEPTS

• **Computing methodologies** → **Natural language processing**; **Probabilistic reasoning**.

KEYWORDS

textual entailment, natural language processing, information extraction

ACM Reference Format:
Kanika Madan, John Hudzina, Thomas Vacek, Frank Schilder, and Tonya Custis. 2019. Textual entailment using word embeddings and linguistic similarity. In *Proceedings of COLIEE 2019 workshop: Competition on Legal Information Extraction/Entailment (COLIEE 2019).* ACM, New York, NY, USA, 5 pages.

COLIEE 2019, June 21, 2019, Montreal, Quebec

1 INTRODUCTION

In natural language processing, textual entailment is a very useful directional relationship [8] in which one piece of text entails or follows from another. The entailing and entailed texts are respectively called text and hypothesis. A textual entailment relationship can be positive, negative or neutral.

- In positive entailment, the text entails the hypothesis.
- In negative entailment, the text contradicts the hypothesis.
- In a neutral relationship, the text neither entails nor contradicts the hypothesis.

Textual entailment has applications in Text Summarization, Machine Translation, Question Answering, and Information Extraction.

International Conference on Artificial Intelligence and Law (ICAIL) organized a Competition on Legal Information Extraction and Entailment (COLIEE). One of the tasks in the competition was on finding textual entailment relationships between the paragraph(s) of an existing case and the paragraph containing the decision from a new case. Given a decision Q of a new case and a relevant case R, a specific paragraph that entails the decision Q needed to be identified. Also, multiple paragraphs in R can be relevant to Q and the task required to identify all of them.

The data set for this task consisted of a corpus of case laws. For a given paragraph containing the decision Q from the base case, the corresponding paragraph containing the entailment relationship in the relevant case R was provided as annotated in the training data. The corpus also contained only those base case Q and relevant case R pairs that were guaranteed to contain an entailment relationship, thus ensuring that there exists at least one entailment relationship

for every pair of base case and relevant case. In addition to this, full text from the base case and all the paragraphs from the relevant case were also provided.

In this paper, we discuss how we approached this task using a ranking methodology to pick the paragraph which is most likely to contain the entailment relationship. Using dependency analysis to find textual entailment has been studied before [7]. For this task, we used a combination of word embedding features and text similarity features to generate these scores and rank the paragraphs.

We also provide a comparison with other methods that we tried in this task, and present how the ranking approach outperformed those. Concretely, we experimented with using supervised machine learning methods to solve this task, and compared the performance of the ranking approach and supervised methods using k-fold cross validation. Our ranking approach methods outperformed the supervised approach methods on both validation set and test set.

2 METHODOLOGY

This paper explains our approach and results for Task 2 in the Competition on Legal Information Extraction and Entailment (COLIEE) 2019 which aimed at finding the entailment relationships between the base case and the relevant cases. The training set consisted of 181 base case paragraphs, and the corresponding paragraph(s) from the relevant case containing the entailment relationship were provided.

We treated this task as a ranking task which provides a score to all of the paragraphs from a relevant case and then ranking these paragraphs using this score, such that higher the score, better is the rank. This score is computed based on the text similarity between the text of the paragraph from the base case Q and the text from the paragraphs of the relevant case. Once all the paragraphs from the relevant case have received a score, we used this score to rank them and pick the top k paragraphs.

We used various classes of features, such as sentence similarity measures, natural language processing features, and semantic similarity features based on the two text spans. We have also used word embedding based features. A word embedding is a parameterized function mapping words in some language to high-dimensional vectors, usually 200 to 500 dimensional vectors [1][4][5][6][2]. We have categorized these features into different classes and provide a brief explanation below:

(1) **Positional Features**:
- avg_lemma_word_offset: Create a set of overlapping lemmas occurring in the base case and the relevant case paragraphs. From each word from this set, create an offset of the word indices and take an average of these offsets.
- avg_match_depth: Create a match set of words with common lemmas in the base case and relevant case paragraphs. Take an average of the min depths in the dependency trees for each word in the above match set.
- avg_min3_match_depth: From the depths of common nodes in the feature "avg_match_depth", take three words with min match depth and take an average over these.
- avg_min_tree_dist: Create a match set of words with common lemmas in base case and relevant case paragraphs. For each pair of nodes in this set, create a minimal spanning tree from the dependency tree such that the distance of the nodes of the two words is minimized from the root. For each word pair, find the distance between them by taking sum of distances of each word from the root. Take an average of these distances.
- avg_sym_diff: Average of symmetric difference of tokens in sliding windows from texts of base case and relevant case paragraphs.
- tok_match_score: Max Jaccard similarity between set of all noun phrases and verb phrases from words in texts from base case and relevant case paragraphs.

(2) **Similarity based features**:
- n-gram_lemmatized: Jaccard similarity of lemmatized and tokenized ngrams for the base case and relevant case paragraphs.
- unigram_lemmatized: Jaccard similarity of lemmatized and tokenized unigrams for the base case and relevant case paragraphs.
- char_match_np: Avg/Min/Max Character match scores between nouns and noun phrases for base case and relevant case paragraphs.
- char_match_vp: Avg/Min/Max Character match scores between verb phrases for base case and relevant case paragraphs.

(3) **Embeddings based features**:
- Word_embedding_similarity: Cosine similarity of the average embedding vectors for the base case and relevant case paragraphs (using word2vec pretrained embeddings).

(4) **n-gram overlap based features**
- Jaccard_similarity_unigram: Jaccard similarity of all unigrams from base case & relevant case paragraphs.
- Jaccard_similarity_bigram: Jaccard similarity of all bigrams from base case & relevant case paragraphs.

Parameters: Random Forest	
Number of Trees	5000
Impurity	gini
Max features	"auto"
Mix Bins	unbounded

Table 1: Parameters for Random Forest

Parameters: Gradient Boosting Trees	
Learning Rate	0.1
Number of Estimators	150
Subsample	1.0
Maximum Depth	15

Table 2: Parameters for Gradient Boosting Trees

Approach

We approached this problem in the following two ways:

(1) Supervised classifier: Convert the task into a binary classification task to predict presence and absence of entailment relationship between each pair of the base case paragraph and the candidate paragraph. We built a random forest classifier for this, with the target labels as 0 and 1 for "no entailment relationship" and "presence of an entailment relationship" respectively.
(2) Ranking approach: For each base paragraph, generate a score for every paragraph from the relevant case and use this score to rank these paragraphs. We calculated this score as an average of the feature values described above and then used it to select the top-k paragraphs using a thresholding mechanism - pick up-to k paragraphs if the difference in their scores is within a small threshold (i.e. the scores are close enough and are within a given threshold). We selected the top-most paragraph as this gave a better precision and recall, and hence a better F1-score.

We use 5-fold cross validation to build and evaluate the two algorithms above.

3 EXPERIMENTAL RESULTS

We used a 5-fold cross validation approach with micro averaging to evaluate our algorithms.

For the supervised approach setting, we created multiple data sets of positive and negative instances with different ratios of negative to positive samples. Positive instances are those pairs of paragraphs from base case and relevant case for which we are given an entailment relationship. To sample negative instances, for every paragraph in the base case, we randomly sampled n paragraphs from the corresponding relevant case which are not marked as containing the entailment relationship. We tried different values for n, ranging from $n = 2, 3, 4, 5$. We then trained a random forest classifier and a gradient boosting classifier on each of these data sets and evaluated their performance.
The parameters for the random forest are summarized in Table 1, and for Gradient Boosting Classifier are summarized in Table 2.

For the ranking approach, we combined the different sets of features by averaging across the different classes of feature values, and then summing them together to get a single score for every pair of base case paragraph and relevant case paragraphs. We computed this score for every possible combination of relevant case paragraph for a given base case paragraph. Thus, if we have say, m base cases, such that each base case has a relevant case with j paragraphs, this approach will give us $m*j$ pairs of paragraphs, and we compute a score for each of them. The scores are generated for every paragraph from the relevant case for a given base case, and are are then used to rank the relevant case paragraphs. For every base case paragraph, the highest scoring paragraphs from the relevant case are then selected. We tried top 1 and top 2 because the average number of relevant paragraphs containing an entailment relationship for every base paragraph was in the order of 1.3 in the training set. The flow of this system is outlined in Figure 1.

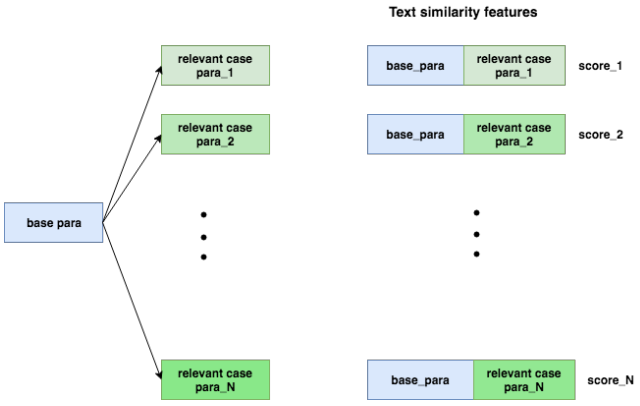


Figure 1: Approach in the ranking methodology

For evaluation, we use Precision (P), Recall (R) and F1-score ($F1$) as following:

$$P = \frac{\#\text{ correctly retrieved paragraphs for all queries}}{\#\text{ retrieved paragraphs for all queries}} \quad (1)$$

$$R = \frac{\#\text{ correctly retrieved paragraphs for all queries}}{\#\text{ relevant paragraphs for all queries}} \quad (2)$$

#Neg/#Pos	Precision	Recall	F1
2	0.28	0.24	0.26
3	0.27	0.20	0.23
4	0.23	0.16	0.19
5	0.17	0.11	0.13

Table 3: Results: 5-fold cross validation from Random Forest

#Neg/#Pos	Precision	Recall	F1
2	0.26	0.25	0.26
3	0.23	0.21	0.22
4	0.22	0.19	0.20
5	0.16	0.13	0.14

Table 4: Results: 5-fold cross validation from GBT

Label	Precision	Recall	F1
0	0.98	0.98	0.98
1	0.53	0.48	0.50

Table 5: Evaluation of ranking approach on training data

$$F1 = \frac{2 \times P \times R}{P + R} \tag{3}$$

We present our results from 5-fold cross validation across different types of systems, and report the precision, recall and F1-scores for each of these. We consistently observed that the ranking approach outperformed the supervised classifier training approach. Also, within the classification approach, we found that the performance of the classifier reduced as the imbalance in the sampled data set increased. The results from 5-fold cross validation from Random Forest and Gradient Boosting Classifier are summarized in tables 3 and 4 respectively.

Our best system from 5-fold cross validation was the ranking approach to pick the top ranked paragraph. This outperformed the other values of n while picking top-n paragraphs. The evaluation of our ranking approach on the training data is presented in table 5, in which labels 0 and 1 represent labels corresponding to the absence and presence of entailment relationships respectively.

The results from this submission are summarized in Table 6. These results were evaluated on the test set and were computed by the COLIEE competition organizers.

Team	Precision	Recall	Fscore
ielab	0.3409	0.3333	0.3371
ielab	0.4545	0.4444	0.4494
ielab	0.2273	0.2222	0.2247
IITP	0.0455	0.0444	0.0449
IITP	0.6591	0.6444	0.6517
IITP	0.7045	0.6889	0.6966
JNLP	0.1364	0.1333	0.1348
JNLP	0.0682	0.0667	0.0674
JNLP	0.5909	0.5778	0.5843
TRCase	**0.6818**	**0.6667**	**0.6742**
TTCL	0.4000	0.8000	0.5333
TTCL	0.3780	0.6889	0.4882
TTCL	0.3882	0.7333	0.5077
UA	0.6364	0.7778	0.7000
UA	0.6296	0.7556	0.6869
UA	0.6538	0.7556	0.7010
UBLTM	0.1182	0.5778	0.1962
UBLTM	0.1273	0.6222	0.2113

Table 6: COLIEE 2019 Task 2 Results on unseen Test Set

4 DISCUSSION AND SUMMARY

The task of finding textual entailment in specialized domains such as legal cases is not an easy one, even for humans, and we found that the task becomes more difficult as the imbalance in the data increases. That is, in an application like this one, in which every base case paragraph usually has one or two paragraphs from the relevant case, finding a suitable approach to find the entailment relationship becomes even more challenging. Supervised approaches might work well on relatively balanced datasets such as SNLI for textual entailment [3]. We found that the ranking approach performed better than a supervised learning approach on a highly imbalanced data set.

REFERENCES

[1] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. 2003. A Neural Probabilistic Language Model. *J. Mach. Learn. Res.* 3 (March 2003), 1137–1155. http://dl.acm.org/citation.cfm?id=944919.944966

[2] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching Word Vectors with Subword Information. *arXiv e-prints*, Article arXiv:1607.04606 (Jul 2016), arXiv:1607.04606 pages. arXiv:cs.CL/1607.04606

[3] Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. *arXiv e-prints*, Article arXiv:1508.05326 (Aug 2015), arXiv:1508.05326 pages. arXiv:cs.CL/1508.05326

[4] Ronan Collobert and Jason Weston. 2008. A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning. In *Proceedings of the 25th International Conference on Machine Learning (ICML '08)*. ACM, New York, NY, USA, 160–167. https://doi.

org/10.1145/1390156.1390177

[5] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. *arXiv e-prints*, Article arXiv:1301.3781 (Jan 2013), arXiv:1301.3781 pages. arXiv:cs.CL/1301.3781

[6] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Doha, Qatar, 1532–1543. https://doi.org/10.3115/v1/D14-1162

[7] Nidhi Sharma, Richa Sharma, and Kanad K. Biswas. 2015. Recognizing Textual Entailment using Dependency Analysis and Machine Learning. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Student Research Workshop*. Association for Computational Linguistics, Denver, Colorado, 147–153. https://doi.org/10.3115/v1/N15-2020

[8] Doina Tatar, Gabriela Serban Czibula, Andreea Diana Mihis, and Rada Mihalcea. 2009. Textual Entailment as a Directional Relation. *Journal of Research and Practice in Information Technology* 41 (2009), 53–64.

A Deep Learning Approach for Statute Law Entailment Task in COLIEE-2019

Nguyen Ha Thanh
nguyenhathanh@jaist.ac.jp
Japan Advanced Institute of Science and Technology

Vu Tran
vu.tran@jaist.ac.jp
Japan Advanced Institute of Science and Technology

Nguyen Le Minh
nguyenml@jaist.ac.jp
Japan Advanced Institute of Science and Technology

ABSTRACT

This paper is about building an end-to-end deep learning approach for statute law entailment problem of COLIEE workshop. In this task, systems are required to answer yes-no questions based on relative clauses found in the Japanese Civil Code. The biggest obstacle in this work is the lack of data. Deep learning requires a large amount of data to automatically extract features.

JNLP team indirectly solve the original problem with a derived problem with more abundant data. Besides, the approach allows us to feed the neural network with shorter sentences, reducing the pressure on its memorization. Through experimental results, we have proven the effectiveness of the method for this problem.

CCS CONCEPTS

• **Computing methodologies** → **Neural networks**.

KEYWORDS

neural networks, deep learning, entailment, legal text, dataset enrichment

ACM Reference Format:
Nguyen Ha Thanh, Vu Tran, and Nguyen Le Minh. 2019. A Deep Learning Approach for Statute Law Entailment Task in COLIEE-2019. In *Proceedings of COLIEE 2019 workshop: Competition on Legal Information Extraction/Entailment (COLIEE 2019).* ACM, New York, NY, USA, 5 pages.

1 INTRODUCTION

COLIEE is an annual workshop aiming for reliable and effective methods to legal information processing. The competition has 4 tasks, of which 2 tasks for case law processing and 2 tasks for handling statute law. In the 2 tasks of statute law, systems need to solve the problem of a person taking the legal bar exam. All lawyer must pass these exams before being admitted to the bar of jurisdictions. Every exam contains yes-no questions requiring the examinees to use their knowledge of the Japanese Civil Code to determine if a statement is true or false. The examinees look for clauses in Japanese Civil Code that related to the statement in the question, then infer from such clauses to give the answer. Based on that thinking process, one task is relevant articles retrieval and the another is entailment.

The problems of the two tasks are described below:

- Problem R (Relevant Articles Retrieval): Given a bar question Q, the system needs to list all articles A relevant to Q.
- Problem E (Entailment): Given a bar question Q and its all relevant articles A, the system needs to answer if A entails Q or not.

with example shown in Table 1.

There could be several approaches to solving problem E. It is usually believed that solving the problem R is the prerequisite for solving problem E. However, this approach has some weaknesses. We explain such weaknesses and propose a different approach in Section 3. The hypothesis we want to prove in this study is the feasibility to use an end-to-end machine learning architecture for the legal reasoning problem.

2 BACKGROUND

2.1 Word embedding

For machine learning approaches, the input of the models are usually numbers, vectors or matrices. As a result, to feed natural language text into the neural network, we need an appropriate representation for them.

There are some methods to convert text into an appropriate numeric representation. Using a number, a random vector or a one-hot vector to represent each word are the simplest ways. However, the relationships between the words are not presented, which leads to poor performance of the model prediction.

The most robust method for representing text is Using word embedding methods. Each word is represented as a vector. In addition, the relationships between the vectors reflect the semantic relationships between the words. Glove [3] and Fasttext [1] are currently the most popular word embeddings.

2.2 GRU Architecture

Gated recurrent unit (GRU) is introduced by Kyunghyun Cho et al. [2]. For GRU architecture, the neural network contains forget gating mechanism so that it is able to prevent exploding and vanishing gradient problems. The following are equations for the forward pass of a GRU unit:

$$z_t = \sigma_g(W_z X_t + U_z h_{t-1} + b_z) \quad (1)$$

$$r_t = \sigma_g(W_z X_t + U_r h_{t-1} + b_r) \quad (2)$$

$$h_t = (1 - z_t) \cdot h_t - 1 + z_t \cdot \sigma_h(W_h X_t + U_h(r_t \cdot h_{t-1}) + b_h) \quad (3)$$

In which, x_t is the input vector, h_t is the output vetor, z_t is the update gate and r_t is the reset gate. W, U and b are the parameters of the model. For $t = 0$, the output vector is $h_0 = 0$. σ_g and σ_h are the activation functions. Originally, σ_g is a sigmoid function and

[1] https://fasttext.cc

COLIEE 2019, June 21, 2019, Montreal, Quebec

Table 1: Example of Retrieval and Entailment Problems

Question	Relevant Articles	Y/N Answer
A special provision that releases warranty can be made, but in that situation, when there are rights that the seller establishes on his/her own for a third party, the seller is not released of warranty.	**Special Agreement Disclaiming Warranty** Even if the seller makes a special agreement to the effect that the seller will not provide the warranties set forth from Article 560 through to the preceding Article, the seller may not be released from that responsibility with respect to any fact that the seller knew but did not disclose, and with respect to any right that the seller himself/herself created for or assigned to a third party.	Y
Usufructuary rights are only established for real estate.	**Establishment of Real Rights** No real rights can be established other than those prescribed by laws including this Code. **Content of Superficies** A superficiary shall have the right to use the land of others in order to own structures, or trees or bamboo, on that land. **Content of Emphyteusis** An emphyteuta shall have the right to engage in cultivation or livestock farming on the land of others by paying rent. **Content of Servitudes** A person entitled to a servitude shall have the right to make lands of others available for the benefit of their own lands in accordance with purposes prescribed in the acts establishing the servitudes; provided, however, that those rights should not violate the provisions (limited to those that relate to public policy) under Section 1 of Chapter 3 (Extent of Ownership).	Y
A manager must engage in management exercising care identical to that he/she exercises for his/her own property.	**Urgent Management of Business** If a Manager engages in the Management of Business in order to allow a principal to escape imminent danger to the principal's person, reputation or property, the Manager shall not be liable to compensate for damages resulting from the same unless he/she has acted in bad faith or with gross negligence.	N

σ_h is a hyperbolic function. Because of its architecture, GRU is able to handle sequence input without vanishing gradient. Besides, it has a good performance with small data sets and can be trained fast with with limited computing resource.

2.3 Texture Entailment Problem

Texture entailment (TE) or Natural language inference (NLI) is a task in NLP. Given a termed text t and hypothesis h, the models need to predict if t entails h (t => h) or not. The answers should be where *yes* or *no*. However, there may be three cases for NLI: t entails h (positive TE), t contradicts h (negative TE) and t does not entail or contradict h (non-TE). The table 2 lists out examples of the three cases.

Table 2: Example of three cases in TE task

Case	Example
Positive TE	*text:* All metals conduct electricity *hypothesis:* Iron is conductive
Negative TE	*text:* All metals conduct electricity *hypothesis:* Iron is non-conductive
Non-TE	*text:* All metals conduct electricity *hypothesis:* Water is non-conductive

Currently, there are several machine learning datasets for this problem. Samuel R. Bowman et al. [1] have introduce the Stanford Natural Language Inference (SNLI) Corpus that contains 570k pairs of written English sentences with three labels *entailment*, *contradiction*, and *neutral*. The Multi-Genre Natural Language Inference (MultiNLI) corpus [4] contains 433k written and spoken sentence pairs obtained by crowd-sourcing.

Table 3: Number of questions for statute law entailment task

Year	ID	New questions	Total questions
-	H18	36	36
-	H19	38	74
-	H20	44	118
-	H21	56	174
-	H22	47	221
-	H23	41	262
-	H24	79	341
2014	H25	60	401
2015	H26	95	496
2016	H27	74	570
2017	H28	77	647
2018	H29	69	716

3 METHOD PROPOSAL

3.1 Data Analysis

Every year, the data set for statute law provided by COLIEE is appended with new questions. For the entailment problem E described in Section 1, the dataset is all the related articles and question from the labeling process in previous year. As a result, the dataset size is limited by the number of questions labeled. The number of questions appended in each year is indicated in the Table 3.

The total number of questions in 2018 is 716. Average and standard deviation of the total number of question are 59.67 and 19.10. There are most questions in H26 (95 questions) and least questions in H18 (38 questions).

Comparing to SNLI and MultiNLI, the total number of questions is extremely small. Figure 1 shows the quantities of examples in COLIEE, SNLI and MultiNLI. It is explainable because of the complication in the legal domain. According to overview of statute law tasks in COLIEE-2018 [5], the number of numbers is not enough for an effectively supervised machine learning.

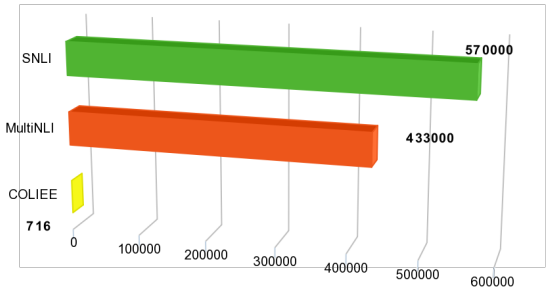


Figure 1: Comparison in quantities of example

We propose a novel approach for overcoming this obstacle. The three features of this approach are problem derivation, data enrichment and article chunking. After that we feed the data represented by fastText word embedding into deep learning architecture of stacked GRU.

3.2 Problem Derivation

The problem R and E described in Section 1 could be viewed as texture entailment perspective. Task R is about eliminating non-related articles of the given question and keeping only articles that entail or contradict the question. Task E determines if the remain articles are in entailment or contradiction relation with the question.

This approach brings some weaknesses in solving the problem E in COLIEE workshop. The first, accuracy of task E depends on the performance of task R. Article elimination in task R could be over, under or wrong elimination, which can affect the prediction accuracy of the model. The second, data fed into the model is not just a pair of sentence but a bunch of long articles. It can lead to variation in length of input. The third, as we have analyzed, the quantity of data is not enough for obtaining a good machine learning model.

Because of the above weaknesses, we have put effort to derive the problem E to another problem. Instead of viewing the answer for questions in Task 4 as the label of texture entailment relation, we could view them as labels for lawful and unlawful statements. The new problem L can be described as: Given a statement S, the system needs to answer if S is lawful or not.

We changed the original entailment problem into a binary classification problem. Instead of using two text phrases as input, the model now only need one statement to predict its label is whether legal or not. There are some labeled statements we got for the new problem L as the following:

- All articles in Japan Civil Code are lawful
- All questions in previous year datasets that are entailed by articles in Japan Civil Code are lawful
- All questions that are not entailed by articles in Japan Civil Code are not lawful

3.3 Input length adjustment

Viewing the articles in Japan Civil Code as a part of our training data, we could enlarge our data set size. There are two parts in the data set, one contains articles in the code and the another contains questions in previous year datasets. However, the problem is, the difference in sentence length distribution of the two parts can lead to poor performance of the trained model because it can challenge the neural network memory ability.

For overcoming this technical problem, we analyze the difference between two parts and apply a method for reducing such difference. The average length of the articles is 43.72 words, with standard deviation is 59.58 while these measurements for the questions are 19.61 and 40.71. The sentences in Japanese Civil Code are averagely longer and more variant than the sentences in the questions in previous year datasets.

We can reduce the average length of articles based on a naive assumption: "Because the article is legal, all of its statements are legal". For each article, instead of adding the whole content into the data set, it is chunked into single statements. For example, article 329 (Order of Priority of General Statutory Liens) could be turn in to two statements as *"(1) In cases where there is conflict among general statutory liens, [...] follow the order listed in each item of*

Table 4: Rules applied for negation statement generation

Original Statement	Negation Statement Generation
contains *not*	Remove *not* from original statement
contains *shall*	Replace *shall* with *shall not*
contains *should*	Replace *should* with *should not*
contains *may*	Replace *may* with *may not*
contains *must*	Replace *must* with *must not*
contains *is*	Replace *is* with *is not*
contains *are*	Replace *are* with *are not*
contains *will be*	Replace *will be* with *will not be*
contains *can*	Replace *can* with *cannot*
contains *cannot*	Replace *cannot* with *can*
contains *with*	Replace *with* with *without*
contains *without*	Replace *without* with *with*
contains *A*	Replace *A* with *No*
contains *An*	Replace *An* with *No*

Article 306." and *"(2) In cases where there is conflict between a general statutory lien [...] who received the benefit of the same."*

After applying the article chunking method, we have the average value and standard deviation in length of text in the civil code data set is 23.34 and 39.65.

3.4 Negation Generation Rules

The model is expected to classify an input statement based on its legality. For one legal statement, the system should be able to determine that the corresponding label is *True*. Moreover, it should have the ability to indicate labels of all negations of the original statement are *False*. For an end-to-end machine learning approach, information used by the system to make a decision is stored in the data set. As a result, to obtain an effective machine learning system, the data set needs to be improved.

Negation of examples in the data set is obtained by a set of rules listed in the table 4. After this phase, we obtained a data set, of which the total number of examples is 4748. This data set is then fed into the deep neural network.

3.5 Classification with deep neural network

A deep neural network is obtained and trained for predicting the legality label of input statement. Statements are first tokenized into a list of words, then converted into a list of embedding vectors using pre-trained fasttext embedding. The embedding dimension is 300 and only 100K most popular words are converted, the other words are represented as a random vector standing for *unknown word*.

In order to train the model by the examples in batches, we need to make the examples the same length. Figure 2 shows the length distribution over all examples in the dataset. The longest statement contains 296 words. However, inserting no-meaning padding tokens into all statements to have 296 words in length will make a big challenge for the model to learn. That will lead to huge computation effort with lower effectiveness. There are 4088 statements containing at most 60 words (86.10%). For optimizing the training phase with limited resource, all statements were pruned and padded into 60 words long.

Table 5: Parameters

Parameters	Value
Embedding dimension	300
Hidden dimension	1000
Number of hidden layers	3
Batch size	32
Learning rate	0.0001
Drop out rate	0.3

In the training phase, we use Adam optimizer with Cross Entropy Loss to train the neural network with stacked GRU architecture. The parameters are shown in Table 5

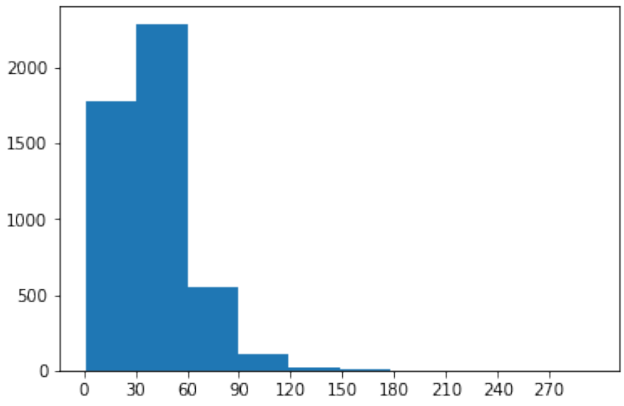


Figure 2: Sentence length variation in the dataset

4 EXPERIMENT AND DISCUSSION

We run the experiment with k-fold cross-validation with $k = 12$. The test data set for each fold is data from previous years. The training set was developed as described in the previous section including statements that were chunked in Civil Japanese Code and the data of the previous year (excluding data of the year in the test set).

Early stopping is applied with $patience = 10$. The results of each fold is shown in the Table 6. The average accuracy is 55.92%, the best accuracy is 64.29% and the lowest accuracy is 51.67%. All result are better than random rate. The difference between the best and the worst accuracy is 12.62%.

We also compare the model accuracy with majority class percentage as the baseline. The average accuracy is 55.88%, higher than the average baseline (54.05%). The Figure 3 demonstrates the difference between the model accuracy and the majority class percentage in each year.

The bar questions are designed differently from year to year and by different questioning techniques. The above result shows that the model not only learned by remembering the training data but also generalized some hidden rules for dealing with unseen data.

In and end-to-end system, the more general the data, the more accurate the prediction. The model is able to perform better when

Table 6: Accuracy with early stopping cross-validation

Year	ID	Loss	Accuracy
-	H18	0.6601	52.7778
-	H19	0.6888	57.8947
-	H20	0.6802	56.8182
-	H21	0.6744	57.1429
-	H22	0.6691	63.8298
-	H23	0.6939	53.6585
-	H24	0.6925	53.1646
2014	H25	0.6897	51.6667
2015	H26	0.6899	54.7368
2016	H27	0.6886	55.4054
2017	H28	0.6879	58.4416
2018	H29	0.6893	55.0725

it is fed with more data with more diverse manner of questioning. This is a possible approach for the next year workshop.

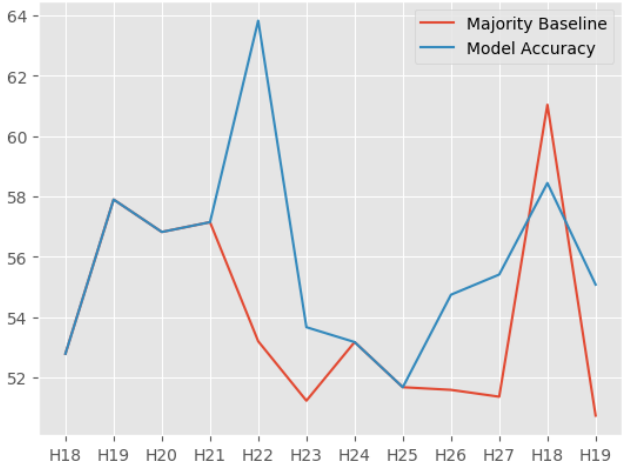


Figure 3: Percentages of model accuracy and the baseline

5 CONCLUSIONS

In this paper, we propose an end-to-end deep learning approach for statute law Entailment Task in COLIEE 2019. We have overcome the challenge for this approach and get the result better than the baseline. Through this work, we have proven that this is possible to apply an end-to-end approach for the problem with limited data. Based on this research, there are some possible directions.

The accuracy of the model on unseen data depends on the data it has already be trained with. The more good data, the better the model can perform. As a result, a transfer learning method is a promising approach for improvement.

The architecture of the neural network in this research only gets one statement as input one by one. However, in the Japanese Civil Code, there are articles related to each other. Graph architecture neural net is also a worth-to-try approach.

REFERENCES

[1] Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. *arXiv preprint arXiv:1508.05326* (2015).

[2] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078* (2014).

[3] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 1532–1543.

[4] Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. Association for Computational Linguistics, 1112–1122. http://aclweb.org/anthology/N18-1101

[5] Masaharu Yoshioka, Yoshinobu Kano, Naoki Kiyota, and Ken Satoh. [n. d.]. Overview of Japanese Statute Law Retrieval and Entailment Task at COLIEE-2018. ([n. d.]).

An approach to Statute Law Retrieval Task in COLIEE-2019

Tran-Binh Dang
School of Information Science
Japan Advanced Institute of Science and Technology
Nomi, Ishikawa, Japan
s1710457@jaist.ac.jp

Thao Nguyen
School of Information Science
Japan Advanced Institute of Science and Technology
Nomi, Ishikawa, Japan
thaod.nguyen@jaist.ac.jp

Vu Tran
School of Information Science
Japan Advanced Institute of Science and Technology
Nomi, Ishikawa, Japan
vu.tran@jaist.ac.jp

Minh Le Nguyen
School of Information Science
Japan Advanced Institute of Science and Technology
Nomi, Ishikawa, Japan
nguyenml@jaist.ac.jp

ABSTRACT

Natural Language Processing (NLP) for legal documents is an emerging field of research. In this paper, we(JNLP) introduce an approach to legal document retrieval, which is a task in Competition on Legal Information Extraction/Entailment 2019. Our method is based on TF-IDF weight and cosine similarity. We experiment with two ways to extract information in legal documents: one employs sentences as a whole and the other uses the only noun and verb phrases.

CCS CONCEPTS

• **Computing methodologies → Feature selection**.

KEYWORDS

TF-IDF, legal document, text chunking, cosine similarity, coliee2019

ACM Reference Format:
Tran-Binh Dang, Thao Nguyen, Vu Tran, and Minh Le Nguyen. 2019. An approach to Statute Law Retrieval Task in COLIEE-2019. In *Proceedings of COLIEE 2019 workshop: Competition on Legal Information Extraction/Entailment (COLIEE 2019).* ACM, New York, NY, USA, 4 pages.

1 INTRODUCTION

Automatic legislation document retrieval has always been a challenging problem due to the complicated nature of legal documents. The complications stem from the use of legal terminologies, polysemous words in law and its tendency for constant amendment. In this paper, we describe a method to deal with legal document retrieval that focuses specifically on statute law documents.

The remainder of this paper is structured as follows. Section 2 will describe the problem of statute law retrieval. Then we propose our model to tackle such problem in section 3. After that, we evaluate the experimental results in section 4. We then give a conclusion about our work in section 5.

COLIEE 2019, June 21, 2019, Montreal, Quebec

2 THE STATUTE LAW RETRIEVAL PROBLEM

The Statute Law Retrieval Problem[1, 3], which originates from the Japanese bar exam, requires extracting a relevant Japanese Civil Code Articles $S_1, S_2, \ldots, S_n$ from the entire Civil Code given a question Q. Relevant articles are ones that can be used to answer the question such that: **entails**(S_1, S_2, ..., S_n , Q) or **entails**(S_1, S_2, ..., S_n, not Q).

Table 1 shows some examples of statute law retrieval problem.

3 THE APPROACH

To solve this problem, we use the approach described in Figure 1. There are three steps include: Pre-processing, Indexing , Scoring and Analysis.

Pre-process: In this step, we normalize words in sentence follow 3 steps:

- Removing stop-words in sentence.
  Stop-words are the most common words in a language. They convey no semantic information and can introduce noises in analysis process.
  There is no universal list of stop-words. Each natural language processing tool usually comes with their own list of stop-words. In our experiment, we used the list from NLTK.corpus.
  Example: (Article: Causes of Dissolution of Juridical Person)
  Input:
  (1) A juridical person shall be dissolved because of
  (i) the occurrence of any cause of dissolution provided in the articles of incorporation or act of endowment, as the case may be;
  (ii) the successful consummation of the business which is the purpose of the juridical person, or the impossibility of such successful consummation;
  (iii) the ruling to commence bankruptcy procedures; or
  (iv) the rescission of the permission of the establishment.
  (2) In addition to the causes listed in the respective items of the preceding paragraph, an incorporated association shall be dissolved because of:
  (i) the applicable resolution of the general meeting; or
  (ii) the attrition of all members.
  Output:
  (1) A juridical person shall dissolved

Table 1: Example of problem

ID	Query	Relevant article
H28-35-2	An owner of the land shall not assume an obligation to a superficiary to make the land available, but a lessor shall assume an obligation to a lessee to make the land available.	**article id="606" name="Repairs of Leased Things"** (1) A lessor shall assume an obligation to effect repairs necessary for using and taking the profits of the leased Things. (2) The lessee may not refuse if the lessor intends to engage in any act that is necessary for the preservation of the leased Thing.
H29-1-A	A is a nineteen-year-old male and subject to the parental authority of his parents. In the case where A concluded a contract for sale without the consent of the persons who had parental authority in relation to him after he had been permitted to carry on business by the persons who had parental authority in relation to him, even if the contract for sale is not concerned with such business, A may not rescind the contract for sale.	**article id="6" name="Permission for Minors to Carry on Business"** (1) A minor who is permitted to carry on one or more kinds of business shall have the same capacity to act as a person of the age of majority as far as such business is concerned. (2) In the case set forth in the preceding paragraph, if the minor may be unable to perform the relevant business for any reason, his/her statutory agent may revoke or limit permission in accordance with the provisions of Part IV (Relatives). **article id="5" name="Juristic Act of Minors"** (1) A minor must obtain the consent of his/her statutory agent to perform any juristic act; provided, however, that, this shall not apply to an act merely intended to acquire a right or to be relieved of a duty. (2) A juristic act in contravention of the provision of the preceding paragraph may be rescinded. (3) Notwithstanding the provision of paragraph 1, in cases the statutory agent permits the disposition of property by specifying the purpose thereof, a minor may freely dispose of the same to the extent of such purpose. The same shall apply in cases his/her statutory agent permits the disposition of the property without specifying any purpose.
H18-9-2	Statutory real rights granted by way of security exist, but statutory usufructuary rights do not exist.	**article id="295" name="Content of Rights of Retention"** (1) If a possessor of a Thing belonging to another person has a claim that has arisen with respect to that Thing, he/she may retain that thing until that claim is satisfied; provided, however, that this shall not apply if such claim has not yet fallen due. (2) The provisions of the preceding paragraph shall not apply in cases where possession commenced by means of a tortious act. **article id="303" name="Content of Statutory Liens"** A holder of a statutory lien shall have the rights to have his/her own claim satisfied prior to other obligees out of the assets of the relevant obligor in accordance with the provisions of laws including this Act. **article id="388" name="Statutory Superficies"** In cases where land and a building on the land belong to the same owner, if a mortgage is created with respect to that land or building, and the execution of that mortgage results in the creation of different owners, it shall be deemed that a superficies has been created with respect to that building. In such cases, the rent shall be fixed by the court at the request of the parties.

(i) occurrence cause dissolution provided articles incorporation act endowment, case may be;
(ii) successful consummation business purpose juridical person, impossibility successful consummation;
(iii) ruling commence bankruptcy procedures;
(iv) rescission permission establishment.
(2) In addition causes listed respective items preceding paragraph, incorporated association shall dissolved of:
(i) applicable resolution general meeting;
(ii) attrition members.

- Stemming. In NLP, stemming is an important step which reduces inflected (or sometimes derived) words to their word stem base or root form (generally a written word form).
Example:(Query "H30-1-U")
Input:
A father may not affiliate his unborn child, even if the mother's consent is obtained
Output:

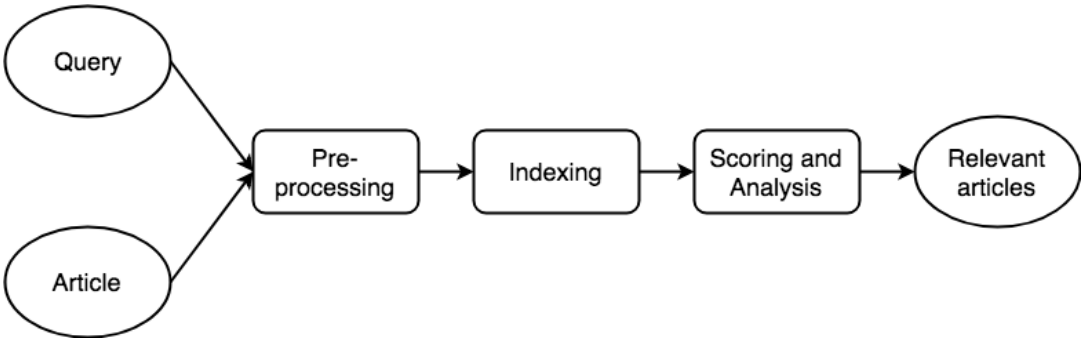


Figure 1: The approach of system

A father may not affiliate his unborn child, even if the mother's consent is obtain

- Chunking. Noun and verb play significant roles in understanding the semantic meaning of the legal article. However, stand-alone nouns and verbs are usually less informative compared to noun and verb phrases. Hence, we used phrase chunking instead of part of speech tagging. For task 3, each sentence was chunked to extract a noun phrase and a verb phrase[2, 5].
  Example:(Article: Juridical Person under Liquidation)
  Input:
  A dissolved juridical person is deemed to still continue to exist to the extent of the purpose of the liquidation until the conclusion of such liquidation.
  Output:
  List of Noun phrase: 'A dissolved juridical person', 'the extent', 'the purpose', 'the liquidation', 'the conclusion', 'such liquidation'.
  List of Verb phrase: 'is deemed', 'continue', 'exist'.

Indexing: We used N-gram and TF-IDF to represent sentence as numerical vector[4]. The most commonly used N-gram are unigram, bigram, and trigam. N-gram can be used in conjunction with TF-IDF weight where the weight is calculated for each n-gram. We indexed the articles using the following methods:

- Based on all content of article
- Based on title of article
- Based on Noun-phrase: Articles were chunking and we only calculate TF-IDF weight for noun-phrases.
- Based on Verb-phrase: Articles were chunking and we only calculate TF-IDF weight for verb-phrases

Scoring and Analysis: We computed cosine similarity between one query vector and all article vectors.

- **cosine_similar(q,a)**: compute the similar between the query **q** and the article **a**
- **cosine_similar(q_N,a_N)**: compute the similar between noun phrase in the query **q** and noun phrase in the article **a**
- **cosine_similar(q_V,a_V)**: compute the similar between verb phrase in the query **q** and verb phrase in the article **a**

With three formulas, we divide to two ways in experiments . A way used only cosine_similar(q,a) (All content). And other ways used cosine_similar(q_N,a_N) and cosine_similar(q_V,a_V) (NP-VP)

After completing computation process, list of similar score is sorted. We used two way to choose set of relevant articles: top-k and threshold. With top-k, we will choose k article which has the highest score as the relevant article. However, number of relevant article in each query is not constant. And this is a difficult problem in the task. We resolve this problem by threshold. With threshold, we compute the ratio distance between k article in top list of relevant article. a,b and c are similar score. They satisfy the condition: c > b > a

$$R\_score(a, b, c) = \frac{b-a}{c-b}$$

Give 3 relevant articles in list: a_{i-1}, a_i and a_{i+1}. Based on the method above, there are 4 elements:

- Distance_score(a_{i+1}, a_i): the difference from similar score of article a_{i+1} and a_i
- Distance_score(a_i, a_{i-1}): the difference from similar score of article a_i and a_{i-1}
- R_score(a_{i-1}, a_i, a_{i+1}): the ratio of distance score between 3 relevant articles a_{i-1}, a_i and a_{i+1}
- Threshold t

After having the elements, we give decision follow two case:

$$Relevant\_articles = \begin{cases} a_{i+1} \Leftrightarrow R\_score(a_{i-1}, a_i, a_{i+1}) < t \\ a_i, a_{i+1} \Leftrightarrow R\_score(a_{i-1}, a_i, a_{i+1}) \geq t \end{cases}$$

4 EXPERIMENT AND RESULTS

We use data from H18 - H29 in this competition. Some information of dataset is show in table 3. The dataset is used to choose model and parameter. We focus on choose threshold t. Based on dataset of previous years, the choosing threshold show in figure 2. Average of number relevant articles per a query is 1.38 article. So, we consider top-3 article is the most relevant and used threshold to choose. With each query, the number relevant articles which was return can be one or two. See the figure 2, the threshold t = 0.1 had the best F2-score. The greater the ratio of distance similar score, the worse the result. In data of "H30" we also used t = 0.1 to choose result.

Table 2: Result of submission

Name	F2	Prec	Recall	MAP
All content	0.5343	0.4592	0.5820	0.5982
NP-VP	0.5054	0.4031	0.5616	0.5750

Name	MAP-L	R_5-L	R_10-L	R_30-L	R_100-L
All content	0.5982	0.6529	0.6860	0.7686	0.8843
NP-VP	0.5750	0.5950	0.6529	0.7686	0.8843

Table 3: Basic statistics of the datasets used in experiment

ID	Number of Query	Average of number relevant articles
H18	36	1.42
H19	38	1.32
H20	44	1.36
H21	56	1.25
H22	47	1.13
H23	41	1.44
H24	79	1.32
H25	60	1.27
H26	95	1.49
H27	74	1.54
H28	77	1.53
H29	69	1.35
Total	716	1.38

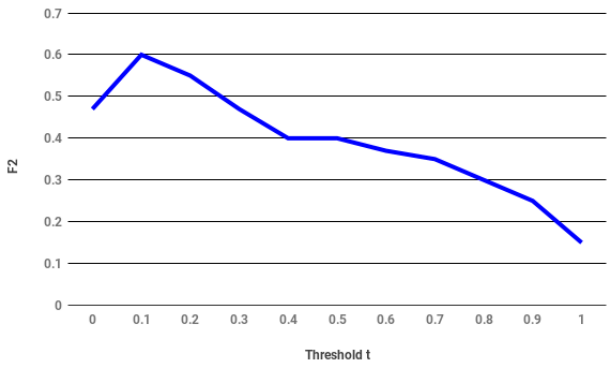


Figure 2: With top-3 relevant article in list, we choose t = 0.1 as the best threshold for experiment

With "Indexing" steps, there are two ways to index. A experiment is all article include: content and title. In other experiment, we only use noun phrase and verb phrase in content and title of article. All of them were evaluated by dataset in this year. The result is shown in table 2. With NP-VP, some information loss. It make performance of method was reduced. When using t = 0.1, both of them only have a difference result in R_5-L, R_10-L. R_30-L and R_100-L are the same in two methods.

5 CONCLUSION

In this report, we were introduced our methods for task 3 (statute law retrieval task). Our methods used TF-IDF as feature and compared the similar between two objects by cosine. With two ways approach in "Indexing" steps, there are no more significant difference. In the future, this approach will combine some other features to improve performance.

REFERENCES

[1] [n. d.]. COLIEE 2019. Retrieved Feb 2019 from https://sites.ualberta.ca/~rabelo/COLIEE2019/
[2] [n. d.]. Extracting Information from Text. Retrieved Apr 2019 from https://www.nltk.org/book/ch07.html
[3] Naoki Kiyota Masaharu Yoshioka, Yoshinobu Kano and Ken Satoh. 2018. Overview of Japanese Statute Law Retrieval and Entailment Task at COLIEE-2018. *International Workshop on Jurisinformatics (JURISIN 2018)* (2018).
[4] Son Truong Nguyen Vu Duc Tran and Minh Le Nguyen. 2018. JNLP Group: Legal Information Retrieval withSummary and Logical Structure Analysis. *International Workshop on Jurisinformatics (JURISIN 2018)* (2018).
[5] Song Z Yoshioka, M. 2018. Hukb at coliee2018 information retrieval task. *International Workshop on Jurisinformatics (JURISIN 2018)* (2018).

Threshold-Based Retrieval and Textual Entailment Detection on Legal Bar Exam Questions

Sabine Wehnert
sabine.wehnert@ovgu.de
Otto von Guericke University Magdeburg
Germany

Sayed Anisul Hoque
sayed.hoque@st.ovgu.de
Otto von Guericke University Magdeburg
Germany

Wolfram Fenske
wolfram.fenske@ovgu.de
Otto von Guericke University Magdeburg
Germany

Gunter Saake
saake@iti.cs.uni-magdeburg.de
Otto von Guericke University Magdeburg
Germany

ABSTRACT

Getting an overview over the legal domain has become challenging, especially in a broad, international context. Legal question answering systems have the potential to alleviate this task by automatically retrieving relevant legal texts for a specific statement and checking whether the meaning of the statement can be inferred from the found documents. We investigate a combination of the BM25 scoring method of Elasticsearch with word embeddings trained on English translations of the German and Japanese civil law. For this, we define criteria which select a dynamic number of relevant documents according to threshold scores. Exploiting two deep learning classifiers and their respective prediction bias with a threshold-based answer inclusion criterion has shown to be beneficial for the textual entailment task, when compared to the baseline.

CCS CONCEPTS

• **Information systems** → **Question answering**; *Similarity measures*; *Relevance assessment*; • **Computing methodologies** → Neural networks.

KEYWORDS

legal text retrieval, textual entailment, stacked encoder, explainable artificial intelligence, threshold-based relevance scoring

ACM Reference Format:
Sabine Wehnert, Sayed Anisul Hoque, Wolfram Fenske, and Gunter Saake. 2019. Threshold-Based Retrieval and Textual Entailment Detection on Legal Bar Exam Questions. In *Proceedings of COLIEE 2019 workshop: Competition on Legal Information Extraction/Entailment (COLIEE 2019).* ACM, New York, NY, USA, 9 pages.

1 INTRODUCTION

Nowadays, globalization poses a challenge for many international organizations, since they need to ensure compliance to laws of all jurisdictions falling under the scope of their activities. Tracking changes in law is a challenging task, especially in statutory law where a single modification may affect the applicability of several legal articles, due to implicit co-dependencies between these documents. While domain experts are mostly required to ensure a reliable assessment of relationships among laws and their implications, the amount of legal documents is hard to oversee for a single person. Therefore, a decision support system can help in finding relevant laws and applying them to a specific question or statement. Finding out whether a statement is true, given a corpus of legal text, falls under the task of legal question answering. A legal question answering system consists of two major parts: document retrieval and textual entailment recognition. In the retrieval phase, relevant law articles are selected for a query, having the form of a statement which shall be supported or contradicted by the law articles from the document collection. During the textual entailment phase, the query and accordingly retrieved legal documents are processed by a classification algorithm which returns "yes" in case of positive textual entailment or "no" otherwise. This work is a contribution to the Competition on Legal Information Extraction/Entailment (COLIEE) competition which provides a dataset from Japanese bar exam questions (translated to English) for evaluating the system performance on both tasks, retrieval and entailment classification. Our contribution involves the following methods:

- We combine results from BM25 scoring with word embedding-based retrieval.
- We develop a stacked encoder ensemble for entailment detection.
- We use thresholding for both approaches.

The remainder of this work is structured as follows: Section 2 outlines related work for both tasks with respect to their achievements using similar methods to our approach. In Section 3, we describe basic concepts for string representation in machine learning models, scoring methods and stacked encoders. We explain our approach in detail in Section 4 and show evaluation results in Section 5. After discussing those results, we conclude our findings and mention our future work considerations.

2 RELATED WORK

The related work for our approach is divided in two parts: The legal information retrieval task and the entailment detection task.

COLIEE 2019, June 21, 2019, Montreal, Quebec

The first part consists of approaches using BM25 scoring or word embeddings, as well as similarity thresholding for a retrieval task. We further present deep learning methods, followed by approaches using thresholds for a textual entailment task.

2.1 Legal Information Retrieval

2.1.1 BM25-Based Solutions. In the COLIEE '16 competition, Onodera and Yoshioka apply BM25 scoring for information retrieval with several extensions using query keyword expansion. Their best result was an F-measure of 54.5% [11]. Arora et al. observe the best score with the BM25 scoring method on a different task of legal document retrieval [1], compared to language models and term frequency - inverse document frequency (TF-IDF) weighting. This finding contradicts the previous observations from COLIEE competitions and the FIRE 2017 IRLeD Track, where ranking SVMs [12] or language models [17, 32] performed better than mere BM25 scoring. Despite those observations, BM25 has shown to provide at least competitive results in many cases, so that we consider it as part of our approach.

2.1.2 Word Embeddings. Word Embeddings have proven to be useful in many natural language processing contexts. We outline several works which have used this document feature representation for legal information retrieval. During the COLIEE '18 competition, the SPABS team was able to overcome vocabulary mismatch in some cases using an RNN-based solution with Word2Vec embeddings trained on English legal documents [34]. Team UB used word embeddings with PL2 term weighting [34]. Yoshioka et al. suggest to use semantic matching techniques for hard questions involving vocabulary mismatch combined with more reliable lexical methods for easy questions [34]. This is the main motivation for our retrieval system, which incorporates lexical BM25 scoring and word embeddings as a semantic representation, respectively.

2.1.3 Thresholding. Thresholding based on similarity values can improve retrieval results by filtering out low-scoring matches. Islam and Inkpen use similarity thresholds to increase the precision of text matching [10]. Stein et al. also employ thresholds for plagiarized document retrieval [29]. In the COLIEE '18 competition, team UBIRLED use a similarity threshold for filtering out irrelevant case judgments [13]. Nanda et al. select the top-5 matching documents from a topic clustering approach [20]. Given the document with the highest similarity score to the query, they apply thresholding, such that any further document will be incorporated into the result set if the distance to the topmost document is less than 15%. Our approach uses a similar criterion for document inclusion.

2.2 Legal Textual Entailment

2.2.1 Deep Learning Approaches. Deep learning approaches have been used by several authors for entailment detection, starting with an application of a single-layered long short-term memory network (LSTM) for input encoding by Bowman et al. [3]. The encoded features from both texts are concatenated and passed through three 200-dimensional tanh layers to a softmax classifier for predicting the entailment relationship. The task is performed on the SNLI[1] dataset which is based on image captioning. Rocktäschel et al. apply neural attention [2] for entailment recognition on the same SNLI corpus [26]. Two LSTM networks are employed for encoding the query and the document, whereby the output vectors from the document are used by an attention mechanism for each word in the respective query. Their method achieves 83.5% accuracy, which is compared to the results by Bowman et al. an improvement of 3.3 percentage points. Liu et al. use a bidirectional LSTM with an attention mechanism [16] and obtained 85% accuracy on the SNLI dataset. A stacked encoder architecture developed by Nie and Bansal achieved 86.1% accuracy on the SNLI dataset. Considering that result as the state-of-the-art, we adapt the main idea to our task in the legal domain and further explain this architecture in section 3.3. Do et al. use a convolutional neural network (CNN) with word embeddings [6]. They incorporate additional features from a TF-IDF and latent semantic indexing (LSI) representation of the sentences. Finally, they feed these features in conjunction with the output of the CNN model into a multi-layer perceptron (MLP) network to predict the answer.

We are inspired by the work of the Chen et al., which focuses on a factoid question and answering system [5]. Their goal is to predict a sequence in the document to answer the query, as opposed to our task of detecting an entailment relationship. They trained two multi-layer bi-directional LSTMs to encode the articles and the query. For encoding the article, they extract multiple features from the query and document pairs: word embeddings of the document (300-dimensional Glove embeddings), an exact matching flag, token features (part-of-speech tags, named entity tags, normalized term frequencies) and attention scores for the similarity of a document and the aligned query. These features are concatenated to form the input vector for the LSTM that encodes the article. The question is encoded without extracting any features. Their evaluation is based on the top five pages returned by the algorithm, and results in 77.8% of correct answers on the SQUAD [23] dataset.

Nanda et al. apply a hybrid network of LSTM networks coupled with a CNN, with the final prediction based on a softmax classifier [20]. They use pre-trained general-purpose word embeddings from the Google news corpus, consisting of 3 billion words. Their accuracy for the COLIEE '17 competition was 53.8%, which they attribute to the general-purpose embeddings which may not capture important semantic relationships needed for the legal domain.

From these works, we conclude that LSTM architectures are suitable for entailment detection for open-domain tasks. However, the COLIEE dataset poses a challenge for deep learning models due to the specific meaning of terms in the legal domain and the rather small size of the dataset. Therefore, we refrain from training word embeddings on the statute law competition corpus only, but consider using other general-purpose word embeddings and a slightly different architecture compared to the previous work. We also find in the related work that extracting additional features from the documents can improve the classifier performance.

2.2.2 Thresholding. Thresholding for the entailment task is applied in two cases: First, the entailment detection can be done by using a similarity threshold. This works similar to an attention layer in a neural network, which applies a focus on a subsection of the input. The second case refers to thresholding in classifier output probabilities. During the COLIEE '15 competition, Kano obtained

[1] nlp.stanford.edu/projects/snli/

good results using a threshold for snippet scoring of the entailment task in the runs `Kanolab1`, `Kanolab2` and `Kanolab3` [12]. Ha et al. show that similarity scoring can be used for entailment detection [9], however, this may only work for low-level inferences. Instead, Carvalho et al. suggest to use classifiers with a rich feature set [4]. We incorporate an exact matching component for obvious cases of positive entailment in our solution and train a classifier for the majority of cases. Glickman et al. tune their probabilistic entailment detection method with a threshold [8]. Rooney et al. note that for entailment detection, exploiting individual classifier bias with a voting scheme or stacking is common [27]. We employ a voting scheme which is based on using thresholds on the output probabilities of classifiers in our ensemble and consider their respective bias.

3 BACKGROUND

In this section, we introduce the required concepts to understand the components of our approach. First, we describe two text representation approaches. Second, we present scoring methods for the retrieval task. Finally, we explain a method for sequence encoding and decoding which is frequently applied in open-domain question answering.

3.1 Text Representation

In order to apply machine learning techniques on text data, a suitable representation of the input is needed. We consider two alternative approaches: the bag-of-words model and word embeddings. The bag-of-words representation collects all words of a document regardless of their order as assigns each distinct word to a unique index. For two texts to be considered as similar, there needs to be a significant lexical overlap. In contrast, word embeddings are semantic representations of a word, which do not require a lexical overlap. However, these representations need to be created, for instance from a neural network which is trained on a reference corpus to predict the next probable word in a sequence. A model for training word embeddings is Word2Vec [19]. The Word2Vec algorithm learns the word embeddings by using the continuous bag-of-words (CBOW) model or the continuous skip-gram model. The embeddings are learned in the CBOW model by predicting the current word based on a context window. In contrast, the continuous skip-gram model predicts the surrounding context given a reference word. General-purpose pre-trained word embeddings are often used, for instance Glove embeddings [22]. During the embedding training, the hidden layers of this network capture information such as the co-occurrence of two words which are present in the same context. The resulting n-dimensional weights from the hidden layer of the network are then called word embeddings. It is important in this context to choose a good reference corpus which represents the vocabulary and common word use in the domain of the task well to obtain adequate embedding weights.

3.2 Weighting and Scoring Methods

Regardless of the chosen representation method, there can be the need to adjust the weight of each word while aggregating to the document level. A common approach for weighting is a discounting of the term frequency within a document by its overall frequency within the corpus. This term frequency - inverse document frequency (TF-IDF) weighting scheme takes into account that only certain words describe a document well. That is, words which occur frequently within a document, but relatively seldom across the rest of the document collection are considered to be keywords and obtain the highest weight. Likewise, words which occur often in the whole corpus are treated as stopwords and their weight is diminished by the denominator term of the inverse document frequency. Generally, retrieval systems often contain a ranking function, where the similarity or a relevance score between the query and candidate documents is computed in order to determine document relevance. We describe two scoring methods in more detail: BM25 similarity scoring and Word Centroid Distance, whereby the latter treats the analogous problem of distance measurement for similarity scoring. BM25 scoring, also referred to as Okapi BM25, is similar to TF-IDF weighting, however it limits the influence of the term frequency and common words between query and document, following a notion of term eliteness as a poisson distribution [25]. The Word Centroid Distance (WCD) has been specifically developed for word embeddings by Kusner et al. [15]. It depicts the minimum cumulative distance that the words of a document need to travel to reach all words of another document in the embedding space. Thereby, a semantic distance metric is provided, given that word embeddings exist which capture those semantic relationships between the words of the respective domain. As a result, paraphrasing documents which convey the same meaning without sharing a single word can be still detected as a match.

3.3 Sequence Encoding and Decoding

Using sentences or article paragraphs as an input for a machine learning algorithm requires a transformation to a lower-dimensional feature representation. Since the order of words and thereby also dependencies among words can impact the entailment relationship, we briefly introduce an approach to model text sequences: the sequence-to-sequence (Seq2Seq) model by Sutskever et al. [31]. It consists of a recurrent neural network (RNN) as an encoder, which maps the input to a fixed-length context vector. This context vector is then accessed by the decoder RNN to generate the target sequence. This approach is popular in the general question answering domain [24]. The Seq2Seq model has been extended by Bahdanau et al. by an attention layer, so that a smaller segment of the fixed-length vector can be used by the decoder to predict the target [2]. Neural network architectures built for question answering encode the query and the document separately. A common practice is to concatenate both context vectors to a single context representation and separate them by markers of question start and sequence end for the final prediction [30]. The question answering problem we consider in this work is reduced to a binary output of confirming or rejecting the statement from the query. Therefore, the decoder is replaced by an entailment classifier, as in the work by Nie and Bansal [21]. While they use an MLP layer followed by a softmax layer for the prediction, we employ a multi-layer neural network, with stepwise-linear activation functions, the rectified linear units (ReLUs), due to their accuracy and simplicity at the same time. In particular, recent studies have investigated the approximation capability of ReLUs in hidden layers, and found that

there is a gain in accuracy, although the nonlinearity of weights is given up [28, 33]. Schmidt-Hieber names desirable properties of the ReLU function: First, the projection property of ReLUs can be exploited to propagate a signal over layers without distortion to synchronize smaller subnetworks of varying depth, and second, the upper bound of all network parameters equals one, thus limiting the amplitude of weight updates during training [28]. To conclude, sequence encoding has been applied on several question answering problems and therefore, we consider it for the task of legal bar exam entailment classification. In the next section, we present the details of our approach.

4 THRESHOLD-BASED RETRIEVAL AND ENTAILMENT DETECTION

In this section, we present our methodology to retrieve relevant laws for a given query. Then, we show our approach for classifying the entailment relationship.

4.1 Law Retrieval combining BM25 Scoring and Word Embeddings

Previous competition results suggest that it can be beneficial to combine the advantage of lexical matching (using the bag-of-words approach) with more recent semantic matching techniques (e.g., word embeddings) [13].

4.1.1 Conceptual Approach. Our retrieval system uses lexical BM25 scoring together with Word2Vec embeddings. An overview of the process is shown in Figure 1. During the preprocessing phase, we separate the articles in the Japanese civil law file by using regular expressions. Then, we create an index over all articles and apply the BM25 scoring method. In addition, we implement a separate pipeline for word embedding creation from the German civil code as an auxiliary corpus. Preliminary experiments have shown that solely using Word2Vec on the English translation of the Japanese civil code does not provide a good embedding quality, so we enriched the corpus for the embedding creation by an English translation of the German civil code[2]. With our trained word embeddings, we applied the Word Centroid Distance (WCD) as a scoring function. Further experiments led us to apply TF-IDF weighting on the Word Centroid Distance to increase the number of relevant documents. We find that both methods returned similar results, however, the word embeddings can contribute correct documents in case of vocabulary mismatch and sufficient semantic similarity. Therefore, we introduced similarity thresholding and prioritize the word embedding-based matching documents over the ones which scored high only on BM25. Since the number of relevant documents varies by query, we define criteria for similarity thresholds in both methods to select one or multiple documents, if there is a high similarity to the query. Setting a high similarity threshold results in a high precision, but a low recall. We therefore select all documents with a high similarity automatically as relevant. Additional documents are only included if they fulfill the threshold criteria which we set manually to optimize retrieval performance on the training data. For instance, a top-1 document with high confidence has either a BM25 score that is at least twice as high as the following second document and larger than the length of the query with an added constant of 20, or that document has a word embedding-based similarity of 90%. The length of the query is considered as a criterion for the BM25 results because the lexical overlap should be significant and not fall below 20% similarity score. Further criteria for lower confidence results are based on the same logic, just with relaxed parameters. Depending on the number of already accepted predictions from higher confidence votes, we define criteria for further result inclusion. For example, if there are no predictions from high, medium or low similarity thresholds, the top-1 prediction without any confidence is retrieved from the results of the word embedding-based similarity scoring.

4.1.2 Implementation Details. Our implementation for the retrieval task is based on Python 3 and Elasticsearch 6.4.1. Furthermore we use the library `vec4ir`[3] provided by Galke et al. for word embedding-based retrieval and re-weighting frequent terms with the inverse document frequency before the centroids are calculated for the WCD [7]. The 300-dimensional Word2Vec embeddings are trained with the continuous bag-of-words model and a context window size of 5. We train the embeddings for 700 epochs on the English version of the German Civil Code and then for further 800 epochs on the English version of the Japanese Civil Code. The input string is tokenized using the spaCy[4] library.

4.2 Textual Entailment using an ensemble of Stacked LSTM Encoders

4.2.1 Conceptual Approach. Our approach for entailment detection consists of stacked encoders and an entailment classifier, inspired by the stacked bidirectional LSTM encoder by Nie and Bansal [21]. Since there may be identical sequences of query and document, such easy cases are returned as positive entailment beforehand. The remaining query-document pairs are processed by the stacked encoder approach. An overview of our classification process is shown in Figure 2. As a first step, the queries and articles are preprocessed by tokenizing the sequences. Both have a separate pipeline, since four additional features are extracted from the article. The first three features are obtained by comparing the article with query tokens and storing matches as a binary feature if a match occurs in the original form, in the lower-cased form or in the lemmatized token form, respectively. We also compute the normalized term frequency for each term in the article. The main advantage of using the normalized term frequency is to lessen the effect of high occurrences of a term in a text sequence [18]. Once extracted, these four features along with the tokenized form of the article and the query become the input to the deep learning model. The feature vectors of query and article are passed to the stacked encoders separately. The first layer in the pipeline of the deep learning model is the embedding layer. Due to a few spelling mistakes, some words in the documents could not be mapped to the Glove embeddings. Because of the small corpus size, we defined a dictionary with correct term spellings. If the Glove embedding is not found, the correct surrogate word is looked up, at least for cases we detected in the training

[2] `https://www.gesetze-im-internet.de/englisch_bgb/`

[3] `https://github.com/lgalke/vec4ir/`

[4] `https://spacy.io/`

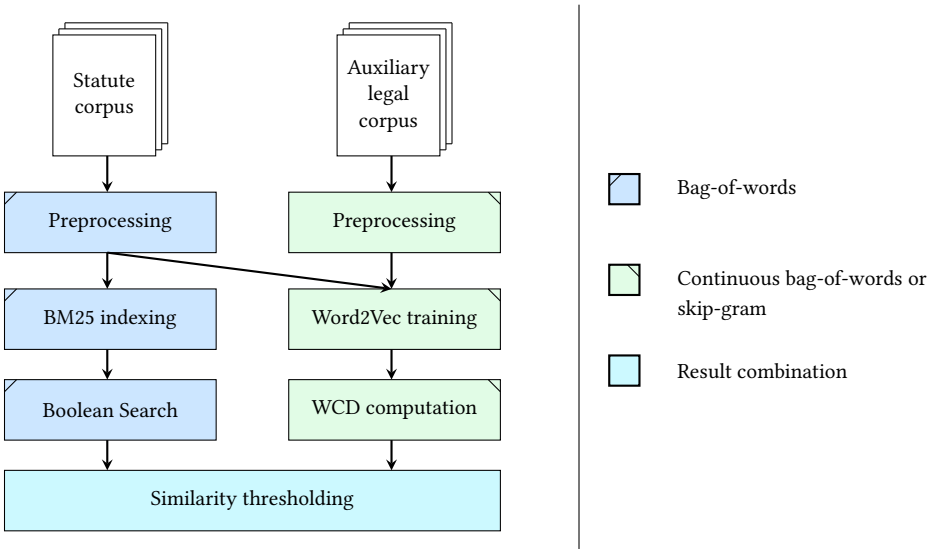


Figure 1: Overview of the information retrieval workflow.

dataset. We then incorporate an attention layer because of the different size of the query and article length (with the article being potentially longer). Another reason for choosing an attention layer is the interpretability of the learned input characteristics. Words causing higher neuron activation can be traced from the attention scores of the model. As a next layer, this network uses a deep bidirectional long short-term memory (LSTM) network and outputs a fixed-length context representation vector. The context vectors from both the article and the query encoder are concatenated and processed by the entailment classifier consisting of stacked linear layers with ReLU activation functions. The binary output is generated by a sigmoid function providing the predicted entailment relationship. Afterwards, models with the highest accuracy are investigated with respect to their bias to set rules for the ensemble voting mechanism. We use the probability of each class - positive or negative entailment - as an indicator of classifier confidence. For example, if one model predicts positive entailment while the other model votes for negative entailment, the positive result is chosen if the probability of the positive class in the approving model is larger than 53%.

4.2.2 Implementation Details. The entailment recognition task is also implemented using Python 3. For preprocessing we again use spaCy. The deep learning models are built by using PyTorch[5]. We train two models using different random seed values for 100 epochs with a batch size of 20 and an Adamax optimizer with the cross entropy loss. There are 3 bidirectional LSTM layers in both encoders - for query and article features - with a hidden size of 100. We apply a dropout of 20% for each bidirectional LSTM layer and use gradient clipping. To generate the prediction, there are 4 stacked linear layers with ReLU activation functions, followed by a sigmoid function for the final output.

[5] https://pytorch.org

5 RESULTS

The evaluation results are based on the assessment of the COLIEE competition. First, we present and discuss the results of the retrieval task. Second, we proceed with the results of the textual entailment task and insights from the assessment. While a complete question answering system is evaluated based on the whole pipeline - retrieval and entailment classification on the retrieved documents - we decided to solve both tasks independently from each other by taking the gold standard set of all relevant documents for the entailment recognition.

5.1 Law Retrieval

We summarize our document coverage for high, medium and low confidence in the training dataset in Table 1. The BM25 scoring method achieves a higher coverage than the word embedding-based solution and has therefore a slightly worse retrieval performance on the high, medium and low confidence thresholds. We decide to prioritize word embedding-based results over the BM25 scoring when the similarity - thus also the confidence - is high. For the combined result on the training dataset (H18-H29), we achieve a precision of 29.3%, a recall of 63.59% and an F2-measure of 48.2%. Preliminary experiments have shown that our approach outperforms simple TF-IDF scoring in Elasticsearch on the training data, as well. Table 2 contains our result in the COLIEE competition compared to the competitors with the highest and the lowest score.

Similarly, on the macro-average evaluation setup, we achieve an F2-score of 46.59%, while other participants scored from 40.08% to 54.93%. However, results from the competition runs indicate that our Word2Vec embeddings cannot guarantee relevant document detection, since our recall at 30 has been outperformed by all other methods. The Word2Vec embeddings may be trained on a larger corpus of legal text - not only civil codes - to provide more valuable semantic representations. Preliminary experiments of using the skip-gram model instead of CBOW for training word embeddings

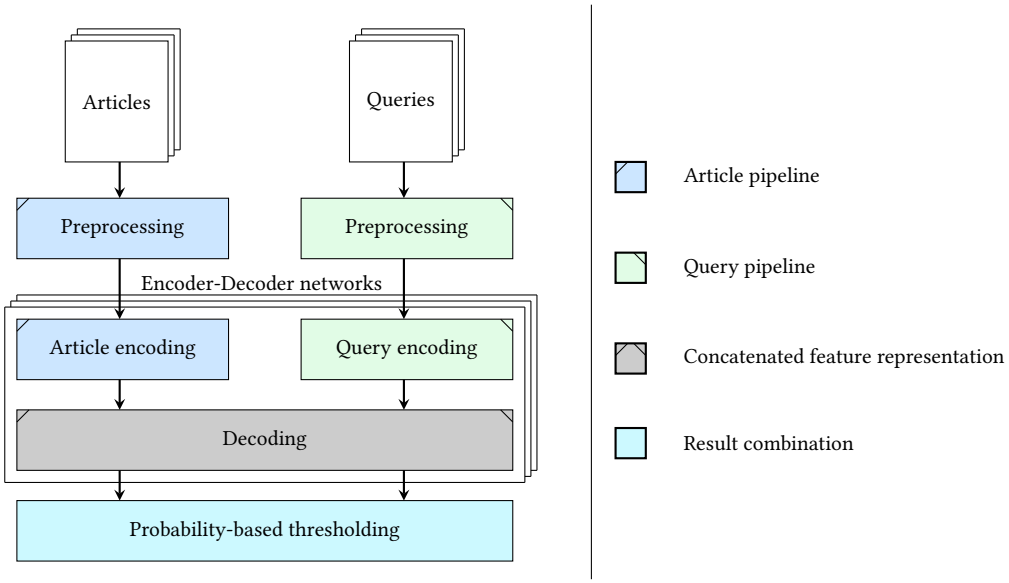


Figure 2: Overview of the textual entailment workflow.

Table 1: Retrieval coverage and performance by using similarity thresholds for several confidence levels. The number of documents retrieved by using BM25-based similarity thresholding is denoted as N_{bm25}, the respective word embedding similarity-based document count as N_{emb}. The metrics coverage *C*, precision *P*, recall *R* and F2-measure *F2* are depicted in percentages.

Metric	High	Medium	Low
N_{bm25}	89	390	453
N_{emb}	53	154	380
C_{bm25}	8.1	35.8	41.7
C_{emb}	4.9	14.0	31.6
P_{bm25}	97.7	87.5	77.6
P_{emb}	100.0	94.9	81.0
R_{bm25}	94.5	79.5	66.3
R_{emb}	94.3	86.9	78.0
$F2_{bm25}$	94.2	79.5	66.9
$F2_{emb}$	94.8	87.2	76.1

decreased all scores on the training dataset, so that we only selected embeddings based on the CBOW model for result submission.

5.2 Textual Entailment

We selected two models with the highest accuracy. They are both biased towards the negative class, although we divided the dataset in such a way that there are more positive examples in the training dataset (with a validation set from H28). We obtain from both models 63.6% accuracy on the validation set. We consider the models to be complementary because they predict positive entailment on different instances. Therefore, we combine their predictions.

Table 2: Retrieval results of our team DBSE for the metrics F2-measure *F2*, precision *P*, recall *R*, mean average precision *MAP*, recall at 5 *R@5*, recall at 10 *R@10* and recall at 30 *R@30*, calculated as macro-averaged values and depicted in percentages.

Metric	UA-TFIDF	**DBSE**	iitptfidf
F2	54.93	**46.59**	40.08
P	59.18	**45.44**	43.88
R	54.42	**49.32**	39.63
MAP	61.81	**51.19**	50.56
R@5	61.98	**51.24**	57.02
R@10	69.42	**61.98**	62.81
R@30	76.03	**66.94**	75.21

Table 3: Entailment classification results of our team DBSE for the accuracy *A* metric, calculated as macro-averaged value and depicted in percentages.

Metric	UA_Ex	**DBSE**	Baseline
A	68.37	57.14	52.04

The assessment from the COLIEE competition shown in Table 3 resulted in an accuracy of our model of 57.14%, compared to the best result of 68.37% and the lowest score of 44.9%. The baseline which always predicts a negative entailment achieved 52.04% accuracy. Since our model works with neural attention, we can visualize the query-document interaction in our model for selected cases. For instance, in task H30-23-A, our model predicted the positive class, although the correct answer is a negative entailment.

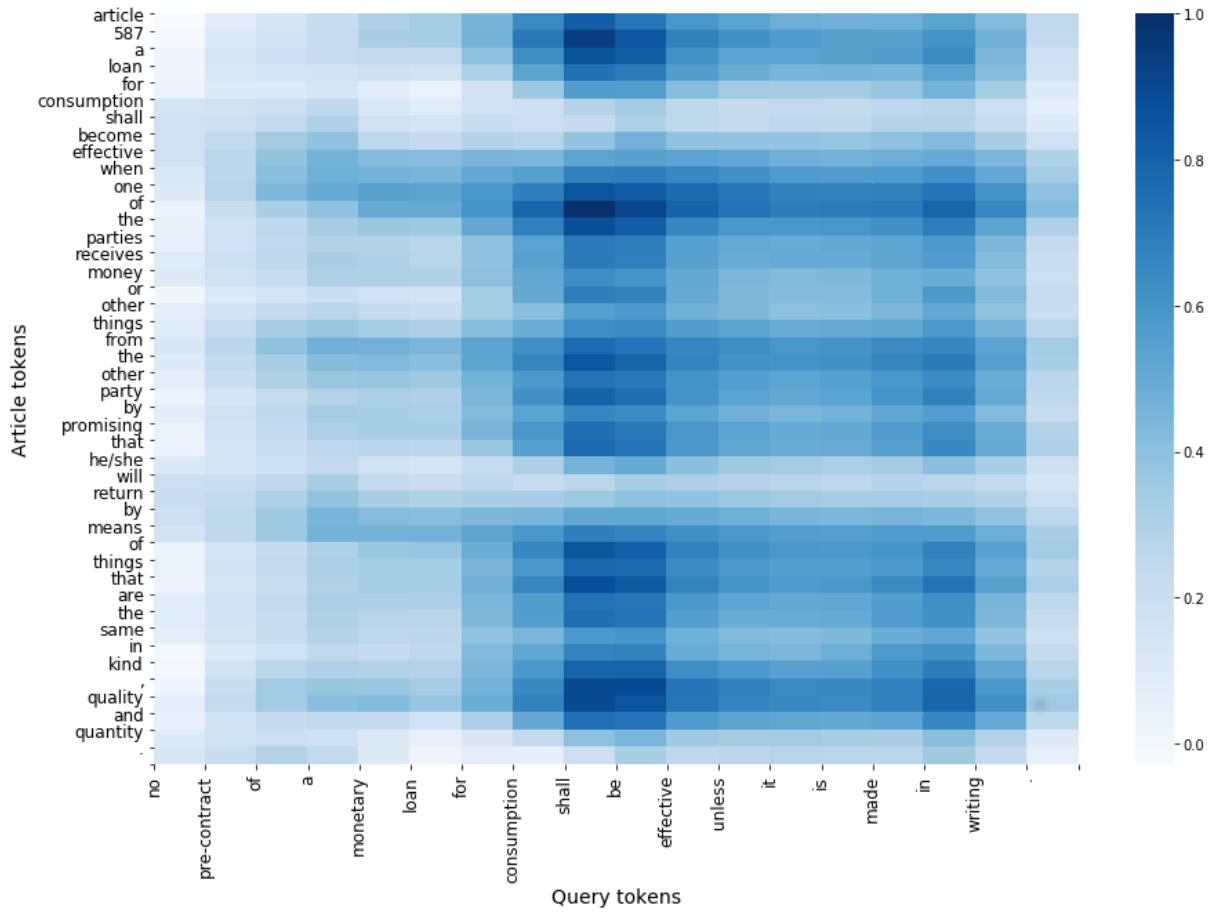


Figure 3: Neural attention example for task H30-23-A. Our model incorrectly predicted a negative entailment relationship to be positive.

Figure 3 illustrates the neuron activation of one of our stacked encoder models. Note that there is almost no neuron activation between the query term "pre-contract" and the article tokens. We suppose that the Glove embeddings do not provide feature information for the legal use of this specific hyphenated term. Interestingly, there is a high neuron activation between the query sequence "shall be effective" and "when one of the parties receives", as well as the article number within the law document. The sequence "that are the same in kind" also exhibits higher activation values for that query sequence. Therefore, we attribute this mistake to the out-of-vocabulary-problem, which is often encountered in domain-specific texts with scarce resources for context modeling with embeddings. While own Word2Vec embeddings were employed for the information retrieval task, we selected Glove embeddings because of the presumably higher likelihood of correctly learned semantic feature representations for negation words and terms such as "must" and "may".

The second example in Figure 4 shows a correctly classified positive entailment relationship. Elevated activation scores are found in the query terms "no successor of the primary obligor may" in conjunction with the article term sequence "no primary obligor, guarantor or successor" and "may make a claim for the extinction". In this case, it can also be observed that the article number has not been attended as strong as in the previous example.

The previous two examples show that our stacked encoder has the ability to attend meaningful inputs, given that the input feature representation is expressive enough to capture the domain-specific context of a word. Due to space limitations, examples with positive predictions were chosen because both encoders are highly biased towards the negative class. In the few cases where they predict a positive entailment, they appear to have a local competence for correctly identifying vital features. Therefore, we expect the performance to increase when word embeddings are generated from a large domain-specific corpus.

6 CONCLUSION

In this work, we present a law retrieval approach and an entailment classifier, developed during the COLIEE '19 competition. For the law retrieval task, we combined BM25 scoring with the Word Centroid Distance for word embeddings, and then applied similarity thresholding for a variable number of retrieved documents per query. Our results outperform some competitor approaches, but we

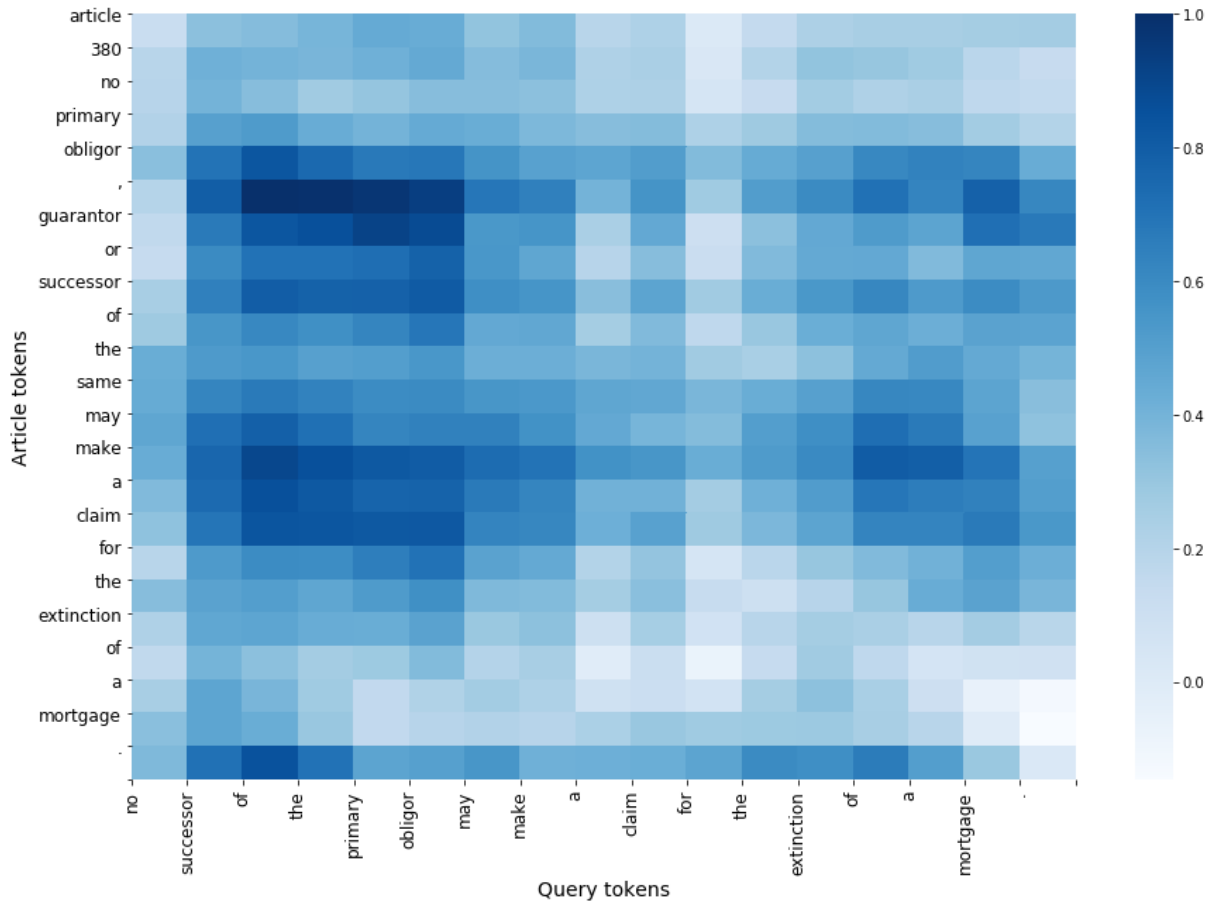


Figure 4: Neural attention example for task H30-13-I. Our model correctly predicted a positive entailment relationship.

expect further improvements by using word embeddings trained on a larger legal corpus. Regarding the textual entailment task, we outperformed the baseline and thereby have shown a benefit of using a stacked encoder architecture with additional features and an attention layer. Future work for the entailment task can be done on additional feature extraction, for instance semantic role labels. Since our ensemble has been selected manually, it can be beneficial to use automated pruning approaches which are designed for imbalanced data to overcome classifier bias and exploit complementary local competences [14]. Another research field is the use of different learning architectures and other word embeddings suitable for the legal domain.

REFERENCES

[1] Piyush Arora, Murhaf Hossari, Alfredo Maldonado, Gareth J.F. Conran, Clare and-Jones, Johannes Paulus, Alexander andKlostermann, and Christian Dirschl. 2018. Challenges in the Development of Effective Systems for Professional Legal Search. In *ProfS/KG4IR/Data: Search@ SIGIR*. CEUR-WS.org, Ann Arbor, Michigan, USA, 29–34.

[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* (2014).

[3] Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Lisbon, Portugal, 632–642. https://doi.org/10.18653/v1/D15-1075

[4] Danilo S. Carvalho, Minh-Tien Nguyen, Chien-Xuan Tran, and Minh-Le Nguyen. 2017. Lexical-Morphological Modeling for Legal Text Analysis. In *New Frontiers in Artificial Intelligence*, Mihoko Otake, Setsuya Kurahashi, Yuiko Ota, Ken Satoh, and Daisuke Bekki (Eds.). Springer International Publishing, Cham, 295–311.

[5] Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading Wikipedia to Answer Open-Domain Questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Vancouver, Canada, 1870–1879. https://doi.org/10.18653/v1/P17-1171

[6] Phong-Khac Do, Huy-Tien Nguyen, Chien-Xuan Tran, Minh-Tien Nguyen, and Minh-Le Nguyen. 2017. Legal question answering using ranking SVM and deep convolutional neural network. *arXiv preprint arXiv:1703.05320* (2017).

[7] Lukas Galke, Ahmed Saleh, and Ansgar Scherp. 2017. Word Embeddings for Practical Information Retrieval. *INFORMATIK 2017* (2017), 2155–2167.

[8] Oren Glickman and Ido Dagan. 2005. Web based probabilistic textual entailment. In *In Proceedings of the 1st Pascal Challenge Workshop*. 33–36.

[9] Quang-Thuy Ha, Thi-Oanh Ha, Thi-Dung Nguyen, and Thuy-Linh Nguyen Thi. 2012. Refining the Judgment Threshold to Improve Recognizing Textual Entailment Using Similarity. In *Computational Collective Intelligence. Technologies and Applications*, Ngoc-Thanh Nguyen, Kiem Hoang, and Piotr Jedrzejowicz (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 335–344.

[10] Aminul Islam and Diana Inkpen. 2008. Semantic text similarity using corpus-based word similarity and string similarity. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 2, 2 (2008), 10.

[11] Mi-Young Kim, Randy Goebel, Yoshinobu Kano, and Ken Satoh. 2016. COLIEE-2016: evaluation of the competition on legal information extraction and entailment. In *International Workshop on Juris-informatics (JURISIN 2016)*.

[12] Mi-Young Kim, Randy Goebel, and Ken Satoh. 2015. COLIEE-2015: evaluation of legal question answering. In *Ninth International Workshop on Juris-informatics (JURISIN 2015)*.
[13] Mi-Young Kim, Yao Lu, Juliano Rabelo, and Randy Goebel. 2018. COLIEE-2018: Evaluation of the Competition on Case Law Information Extraction and Entailment. [online]. https://sites.ualberta.ca/~rabelo/COLIEE2019/COLIEE2018_CL_summary.pdf
[14] Bartosz Krawczyk and Michał Woźniak. 2018. Leveraging Ensemble Pruning for Imbalanced Data Classification. In *2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, 439–444.
[15] Matt J. Kusner, Yu Sun, Nicholas I. Kolkin, and Kilian Q. Weinberger. 2015. From Word Embeddings to Document Distances. In *Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37 (ICML'15)*. JMLR.org, 957–966. http://dl.acm.org/citation.cfm?id=3045118.3045221
[16] Yang Liu, Chengjie Sun, Lei Lin, and Xiaolong Wang. 2016. Learning natural language inference using bidirectional LSTM model and inner-attention. *arXiv preprint arXiv:1605.09090* (2016).
[17] Arpan Mandal, Kripabandhu Ghosh, Arnab Bhattacharya, Arindam Pal, and Saptarshi Ghosh. 2017. Overview of the FIRE 2017 IRLeD Track: Information Retrieval from Legal Documents. In *FIRE (Working Notes)*, Prasenjit Majumder, Mandar Mitra, Parth Mehta Mehta, and Jainisha Sankhavara (Eds.). CEUR-WS.org, Bangalore, India, 63–68.
[18] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA.
[19] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and Their Compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2 (NIPS'13)*. Curran Associates Inc., USA, 3111–3119. http://dl.acm.org/citation.cfm?id=2999792.2999959
[20] Rohan Nanda, Adebayo Kolawole John, Luigi Di Caro, Guido Boella, and Livio Robaldo. 2017. Legal Information Retrieval Using Topic Clustering and Neural Networks. In *COLIEE 2017. 4th Competition on Legal Information Extraction and Entailment (EPiC Series in Computing)*, Ken Satoh, Mi-Young Kim, Yoshinobu Kano, Randy Goebel, and Tiago Oliveira (Eds.), Vol. 47. EasyChair, 68–78. https://doi.org/10.29007/psgx
[21] Yixin Nie and Mohit Bansal. 2017. Shortcut-Stacked Sentence Encoders for Multi-Domain Inference. In *Proceedings of the 2nd Workshop on Evaluating Vector Space Representations for NLP*. Association for Computational Linguistics, Copenhagen, Denmark, 41–45. https://doi.org/10.18653/v1/W17-5308
[22] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Doha, Qatar, 1532–1543. https://doi.org/10.3115/v1/D14-1162
[23] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ Questions for Machine Comprehension of Text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Austin, Texas, 2383–2392. https://doi.org/10.18653/v1/D16-1264
[24] Siva Reddy, Danqi Chen, and Christopher D Manning. 2018. Coqa: A conversational question answering challenge. *arXiv preprint arXiv:1808.07042* (2018).
[25] Stephen Robertson and Hugo Zaragoza. 2009. The Probabilistic Relevance Framework: BM25 and Beyond. *Information Retrieval* 3, 4 (2009), 333–389.
[26] Tim Rocktäschel, Edward Grefenstette, Karl Moritz Hermann, Tomáš Kočiskỳ, and Phil Blunsom. 2015. Reasoning about entailment with neural attention. *arXiv preprint arXiv:1509.06664* (2015).
[27] Niall Rooney, Hui Wang, and Philip S Taylor. 2014. An investigation into the application of ensemble learning for entailment classification. *Information Processing & Management* 50, 1 (2014), 87–103.
[28] Johannes Schmidt-Hieber. 2017. Nonparametric regression using deep neural networks with ReLU activation function. *arXiv preprint arXiv:1708.06633* (2017).
[29] Benno Stein, Sven Meyer zu Eissen, and Martin Potthast. 2007. Strategies for Retrieving Plagiarized Documents. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '07)*. ACM, New York, NY, USA, 825–826. https://doi.org/10.1145/1277741.1277928
[30] Eylon Stroh and Priyank Mathur. 2016. Question Answering Using Deep Learning. [online]. https://cs224d.stanford.edu/reports/StrohMathur.pdf
[31] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to Sequence Learning with Neural Networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2 (NIPS'14)*. MIT Press, Cambridge, MA, USA, 3104–3112. http://dl.acm.org/citation.cfm?id=2969033.2969173
[32] Liuyang Tian, Hui Ning, Leilei Kong, Zhongyuan Han, Ruiming Xiao, and Haoliang Qi. 2017. HLJIT2017@ IRLed-FIRE2017: Information Retrieval From Legal Documents. In *FIRE (Working Notes)*, Prasenjit Majumder, Mandar Mitra, Parth Mehta Mehta, and Jainisha Sankhavara (Eds.). CEUR-WS.org, Bangalore, India, 82–85.
[33] Dmitry Yarotsky. 2018. Optimal approximation of continuous functions by very deep ReLU networks. *arXiv preprint arXiv:1802.03620* (2018).
[34] Masaharu Yoshioka, Yoshinobu Kano, Naoki Kiyota, and Ken Satoh. 2018. Overview of Japanese Statute Law Retrieval and Entailment Task at COLIEE-2018. [online]. https://sites.ualberta.ca/~rabelo/COLIEE2019/COLIEE2018_SL_summary.pdf

Author Index