Two-step Cascaded Textual Entailment for Legal Bar Exam Question Answering

Mi-Young Kim and Randy Goebel
Alberta Machine Intelligence Institute
Department of Computing Science
University of Alberta
Edmonton AB, T6E 2G8 Canada
{miyoung2, rgoebel}@ualberta.ca

ABSTRACT

Our legal question answering system combines legal information retrieval and textual entailment, and exploits semantic information using a logic-based representation. We have evaluated our system using the data from the competition on legal information extraction/entailment (COLIEE)-2017. The competition focuses on the legal information processing required to answer yes/no questions from Japanese legal bar exams, and it consists of two phases: ad hoc legal information retrieval (Phase 1), and textual entailment (Phase 2). Phase 1 requires the identification of Japan civil law articles relevant to a legal bar exam query. For this phase, we have used an information retrieval approach using TF-IDF combined with a simple language model. Phase 2 requires a yes/no decision for previously unseen queries, which we approach by comparing the approximate meanings of queries with relevant statutes. Our meaning extraction process uses a selection of features based on a kind of paraphrase, coupled with a condition/conclusion/exception analysis of articles and queries. We also extract and exploit negation patterns from the articles. We construct a logic-based representation as a semantic analysis result, and then classify questions into easy and difficult types by analyzing the logic representation. If a question is in our easy category, we simply obtain the entailment answer from the logic representation; otherwise we use an unsupervised learning method to obtain the entailment answer. Experimental evaluation shows that our result ranked highest in the Phase 2 amongst all COLIEE-2017 competitors.

CCS CONCEPTS

• **Information systems → Information retrieval**; *Information retrieval query processing* • **Applied computing → Law, social and behavioral sciences;** *Law*

ICAIL '17, June 12-16, 2017, London, United Kingdom
© 2017 Copyright is held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-1-4503-4891-1/17/06...$15.00
http://dx.doi.org/10.1145/3086512.3086550

KEYWORDS

Textual entailment, Question answering, legal text mining, logic representation

ACM Reference Format

Mi-Young Kim and Randy Goebel. 2017. Two-step Cascaded Textual Entailment for Legal Bar Exam Question Answering. In *Proceedings of ICAIL '17, London, United Kingdom, June 12-16 2017*, 6 pages.
https://doi.org/10.1145/10.1145/3086512.3086550

1 SUMMARY OF OUR APPROACH

Our approach to legal question answering combines information retrieval and textual entailment. We achieve this combination with a number of intermediate steps. For instance, consider the question *"Is it true that 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."* A system must first identify and retrieve relevant documents (typically legal statutes), and subsequently, identify those sentences most relevant to answering the question. Finally, it must extract and compare semantic connections between the question and the relevant sentences, and confirm a threshold of evidence about whether an entailment relation holds.

The Competition on Legal Information Extraction/Entailment (COLIEE) 2017[1] focuses on two aspects of legal information processing related to answering yes/no questions from legal bar exams: legal document retrieval (Phase 1), and whether there is a textual entailment relation between a query and relevant legal documents (Phase 2).

We treat Phase 1 as an ad-hoc information retrieval (IR) task. The goal is to retrieve relevant civil law statutes or articles that are related to a question in legal bar exams, from which we can confirm a yes or no answer based on deciding if the question is entailed by the relevant statutes.

We approach the information retrieval part of this problem (Phase 1) with two models, both based on statistical information. One is the TF-IDF model [6], i.e., term frequency-inverse

[1] http://webdocs.cs.ualberta.ca/~miyoung2/COLIEE2017/

document frequency. The idea is that relevance between a query and a document depends on their intersecting word set. The importance of words is measured with a function of term frequency and document frequency as parameters.

Another popular model for text retrieval is language model-based information retrieval [16]. The language modeling approach to IR directly models the idea that a document is a good match to a query if the document model is likely to generate the query, which will in turn happen if the document contains the query words often. This model has shown good performance in the information retrieval field. We use this model to retrieve relevant legal statutes of a query bar exam.

The goal of Phase 2 is to construct yes/no question answering systems for legal queries, by heuristically confirming entailment of a query from relevant articles. The answer to a question is typically determined by measuring some kind of semantic similarity between question and answer. Because the legal bar exam query and relevant articles are complex and varied, we need to carefully determine what kind of information is needed for confirming textual entailment. Here we exploit a logic-based representation that arises from syntactic analysis to produce a semantic result. After extracting features from that logical representation, we train the system to learn models for semantic matching between question and corresponding articles. These feature extraction methods are coupled with negation analysis, and then used to construct an unsupervised model to provide the required yes/no answers. We classify questions into "easy" or “difficult”. If a question is easy, then we obtain the entailment result directly from the logic representation. Otherwise, we perform an unsupervised learning step to obtain the entailment result.

The rest of our paper is structured as follows. First, we explain Phase 1 in Section 2, and then describe Phase 2 in Section 3. We show our experimental setup, results, and error analysis in Section 4. Related work will be given in Section 5. Finally, our future work and conclusions are described in Section 6.

2 PHASE 1: LEGAL INFORMATION RETRIEVAL

2.1 IR Models

We constructed two kinds of information retrieval models: the term frequency–inverse document frequency (tf-idf) model and language model-based information retrieval model. We will describe the two components in the following.

2.1.1 The tf-idf model. One of our baseline models is a tf-idf model implemented in Lucene[2], an open source IR system .

The simplified version of Lucene's similarity score of an article to a query is:

$$tf\text{-}idf(Q,A) = \sum_{t \in Q \cap A} \{\sqrt{tf(t,A)} \times [1 + \log(idf(t))]^2\}$$

Table 1: Performances of Information Retrieval

Method	Precision	Recall	F-measure
TF-IDF	0.6667	0.4727	0.5532
Language model	0.6026	0.4272	0.5000

Table 2: Performances between our method vs. others

Method	Precision	Recall	F-measure
Our TF-IDF	**0.6667**	**0.4727**	**0.5532**
HUKB-1	0.6582	0.4727	0.5503
HUKB-2	0.5870	0.4910	0.5347
HUKB-3	0.5514	0.5636	0.5438
iLis7-1	0.7350	0.5546	0.6321
iLis7-2	0.6547	0.5000	0.5670
JAISTNLP2-1a	0.6282	0.4455	0.5213
JAISTNLP2-1b	0.6154	0.4364	0.5106
JNLP1-R	0.6860	0.5364	0.6020
JNLP1-RT	0.6897	0.5455	0.6091
JNLP1-T	0.5000	0.3545	0.4149
KID17	0.7037	0.5182	0.5969
KIS-IE-M	0.2632	0.2727	0.2679
KIS-IE-NM	0.3462	0.2455	0.2872
NOR17	0.4622	0.5000	0.4803
VNPT	0.4306	0.2818	0.3407

The score tf-idf(Q,A) is a measure which estimates the relevance between a query Q and an article A. First, for every term *t* in query A, we compute tf(*t*,A), and idf(*t*). The score tf(*t*,A) is the term frequency of *t* in the article A, and idf(*t*) is the inverse document frequency of the term *t*, which is the number of articles that contain *t*. The final score is the sum of the scores of terms in both the article and the query. The bigger tf-idf(Q,A), the more relevance between the query Q and the article A.

2.1.2 The language model-based IR. We would like to estimate, the probability of the query Q (legal bar exam question) given the language model of document d (legal statute) as follows.

$$\hat{p}(Q \mid M_d) = \prod_{w \in Q} \hat{p}(w \mid M_d) \times \prod_{w \notin Q} 1.0 - \hat{p}(w \mid M_d) \cdot$$

The first term is the probability of generating words in the query and the second term is the probability of not generating other terms. The specific probabilities for $\hat{p}(Q \mid M_d)$ are defined in Ponte and Croft [16]. We used the Lucene package for this language model-based IR. In the next subsection, we show the experimental results using the two IR models.

2.2 Experiments for Phase 1

The COLIEE legal IR task has several sets of queries with the Japan civil law articles as documents (1044 articles in total). Here follows one example of the query and a corresponding relevant article.

Question: A person who made a manifestation of intention which was induced by duress emanated from a third party may

[2] https://lucene.apache.org/

$1.\ conclusion :=$
$segment_{last}(sentence,\ keyword)$
$2.\ condition :=$
$\sum_{i \neq last} segment_i(sentence,\ keyword)$
$3.\ condition := condition\ [or]\ condition$
$4.\ condition := sub\_condition\ [and]\ sub\_condition$
$5.\ exception\_sentence :=$
$sentence\ including\ exception\_keyword$
$6.\ exception\_condition :=$
$\sum_{i \neq last} segment_i(exception\_sentence,\ exception\_keyword)$
$7.\ exception\_condition\ :=$
$exception\_condition\ [or]\ exception\_condition$
$8.\ exception\_condition\ :=$
$sub\_exception\_condition\ [and]$
$sub\_exception\_condition$
$9.\ exception\_conclusion := NEG\ conclusion$

Figure 1: Rules for component detection

rescind such manifestation of intention on the basis of duress, only if the other party knew or was negligent of such fact.

Related Article: (Fraud or Duress) Article 96(1) Manifestation of intention which is induced by any fraud or duress may be rescinded. (2)In cases any third party commits any fraud inducing any person to make a manifestation of intention to the other party, such manifestation of intention may be rescinded only if the other party knew such fact.(3)The rescission of the manifestation of intention induced by the fraud pursuant to the provision of the preceding two paragraphs may not be asserted against a third party without knowledge.

Before the final test set was released, we received 10 sets of queries for a dry run. The 10 sets of data include 581 queries. For test data, we received 78 queries. They provided original Japanese data English translation. For the Task 1, we used English data as input to the Lucence IR package. The metric for measuring our IR models is Precision, Recall, and F-measure.

Table 1 presents the result of using the different models on the test data. The result shows that the TF-IDF model achieves better performance than the language model-based IR model.

Table 2 shows comparison between our IR result of the TF-IDF model and other system's results on the test data. Our result ranked 6th among the 16 submissions.

3 PHASE 2: ANSWERING YES/NO QUESTIONS

Our procedure of textual entailment to answer Yes/No questions is as follows:

1. Divide a query and the corresponding article into "Condition(s)," "Conclusion," and "Exception-condition(s)."
2. Detect Negation
3. Construct logic representation from the syntactic analysis tree
4. Classify the question into easy or difficult type (two-steps)
5. 1st step: If a question is an easy type, determine the entailment answer based on the logic representation
6. 2nd step: Otherwise, perform an unsupervised learning

We will explain each procedure in the next subsections.

3.1 Condition, conclusion and exceptional cases detection

From our data, we extract components based on the patterns in Figure 1. In case where multiple patterns are triggered, we apply each pattern based on the rule order in Figure 1.

We segment sentences using keywords in the condition. The keywords of the condition are as follows: *"in case(s)," "if," "unless," "with respect to," "when,"* and *", (comma)."* After this segmentation, the last segment is considered to be a conclusion, and the rest of the sentence is considered as a condition. (We used the symbol Σ to denote the concatenation of the segments). We also distinguish segments that denote exceptional cases. Currently, we take the *exception_keyword* indication as "*Provided, however, this shall not apply, if(unless).*"

The following is an example of the condition and conclusion detection:

<Civil law example> A person who employs others for a certain business, shall be liable for damages inflicted on a third party by his/her employees with respect to the execution of that business; Provided, however, that this shall not apply, if the employer exercised reasonable care in appointing the employee or in supervising the business, or if the damages could not have been avoided even if he/she had exercised reasonable care.

(1) Conclusion => shall be liable for damages inflicted
on a third party by his/her employees with respect to the
execution of that business;

(2) Condition => A person who employs others for a certain business

(3) Exception sentence => Provided, however, that this shall not apply, if the employer exercised reasonable care in appointing the employee or in supervising the business, or
if the damages could not have been avoided even if he/she had exercised reasonable care.

Conclusion => NEG shall be liable for damages inflicted
on a third party by his/her employees with respect to the
execution of that business.

Condition
Condition (OR) =>
Condition (OR) => if the employer exercised
reasonable care in appointing the employee
Condition => in supervising the business
Condition => if the damages could not have been

Table 3: Negation types

Negation type	Example
Negation affix	not, no, unless, without, unable
Negation words	rescind, revoke, lack, cease, block
Negation concepts	125(no), 444(cancel)

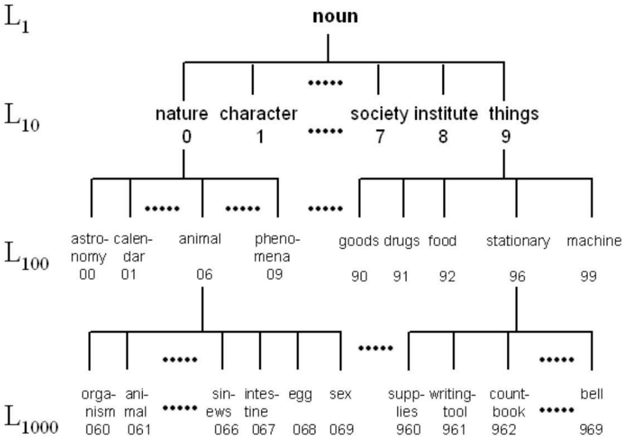


Figure 2: Concept hierarchy of the Kadokawa thesaurus [4]

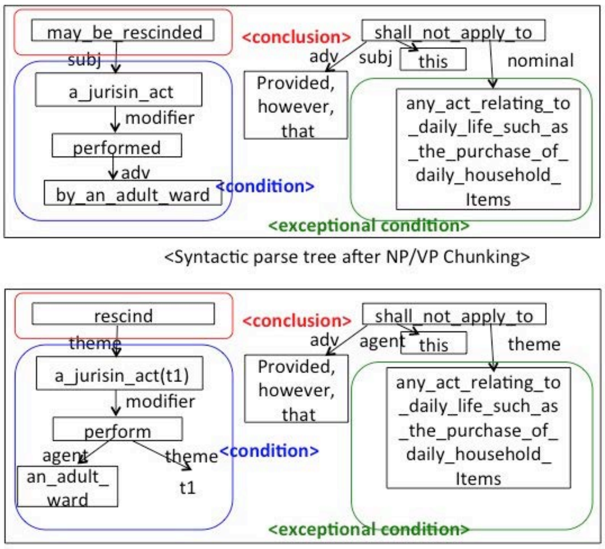
avoided even if he/she had exercised reasonable care.

3.2 Detecting negation

As part of our semantic analysis, we construct a simple negation dictionary. A most important features in determining semantic entailment is the accurate attribution of negation. In our approach, we construct a negation knowledge base as described in Kim et al. [18]. We identify three types of negation expressions as shown in Table 3: one is to note negation prefixes such as ”not,” ”no,” etc. Another is the case where the word itself conveys negative information. To extend our identification of negation words, we also use the Kadokawa thesaurus which has a 4-level hierarchy of about 1,100 semantic classes. Concept nodes in level L1, L10, and L100 are further divided into 10 subclasses, as shown in Figure 2.

3.3 Step 1: Semantic representation

We constructed a logic-based representation from semantic parsing using a dependency parser [7] previously created for agglutinative languages. This parser embeds its own morphological analyzer and uses a dictionary that includes rich linguistic knowledge such as verb-argument information syntactic function and semantic category for each argument. This is a rule-based parser based on rich dictionary knowledge. Because this parser is for the Korean language, we translated the Japanese data into Korean using the Excite translation tool[3]. We find verbs and their corresponding arguments using dependency relations, and assign semantic roles using the verb-argument dictionary and syntactic function. Since a dependency parser structure is very similar to the semantic parse structure, it is relatively easy to transform the dependency parser result into our desired semantic parse structure. We construct a semantic representation by transforming passive forms to active forms in the syntactic parse tree, and determine the semantic role using the Kadokawa thesaurus concept numbers and dictionary information in the syntactic tree structure of Kim et al. [7]. For the case role matching between syntactic roles and semantic roles, we constructed simple heuristic rules based on our own training data. The dictionary in Kim et al. [7] includes 113,000 entries, each of which identify required arguments of a predicate, semantic types, and semantic roles of arguments. Figure 3 is an example of the semantic representation for the sentence “*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.*”



<Syntactic parse tree after NP/VP Chunking>

<Semantic parse tree>

Figure 3: Example of the semantic representation

As shown in Figure 3, a noun phrase chunk (e.g., “*a jurisin act*”) or a verb phrase chunk (e.g., “*may be rescinded*”) are considered single nodes in the tree. In Figure 3, the syntactic parse tree says that ”*a jurisin act*” is a subject of ”*may be rescinded*” and ”*by an adult ward*” is an adverbial of ”*performed*”. However, after converting passive form to active form, ”*a jurisin act*” functions as a theme of ”*rescind*”, and also a theme of ”*perform*”. ”*An adult ward*” can be an agent of the verb ”*perform*”. The parsing dictionary has the information that ”*perform*” requires an agent(concept_no.:n5) and a theme (concept_no.: n3, n4, n6, n7

[3] http://www.excite.co.jp/world

,n8, n9). In the Kadokawa concept number description, 'v' means 'verb', and 'n' means 'noun'. Logical representation is constructed by segmenting the input sentence into the condition, conclusion, exceptional condition, and exceptional conclusion. After that, we extract the Kadokawa thesaurus concept number for the predicate and corresponding arguments in each segment. Dictionary information and corresponding logical representation of the semantic parse tree in Figure 3 are as below.

==

DICTIONARY INFO:
[VERB(Kadokawa#): ARG(required argument with the corresponding concept number)]
1. rescind(v444) : ARG(agent(n5*), theme(n*))
2. perform(v783) : ARG(agent(n5*), theme(n3*,n4*, n6*,n7*,n8*,n9*))
3. apply to(v381) : ARG(agent(n*), theme(n*))
[NOUN(Kadokawa#)]
adult(n124), ward(n507), jurisin(n734), act(n360), rescind(
v444), act(n360), daily life(n350), purchase(n742),
household item(n900)

====================================

LOGICAL REPRESENTATION:
[Condition] v783(AGT:n124,n507, THM:n734,n360)
→**[Conclusion]**v444(THM : n734, n360)
[Exception Condition]n360, n350, n742, n900
→ **[Exception Conclusion]**NEG v444(THM : n734,n360)

====================================

Generally, each statue can have two logical components: one is a general case, and the other is an exceptional case. If a corresponding query belongs to the exceptional case, then we will use the exceptional case logic for the statute and compare it's meaning with that of the query. Otherwise, the logic for general case in the statute will be compared with a query.

3.3.1 Contradiction between logical representations

We augment negation information with the logical form. The following shows the augmented result of the logical representation of Figure 3. Notice that v444 is replaced with NEG.

==

Negation info:
rescind (v444): **NEG**

====================================

AUGMENTED LOGICAL REPRESENTATION:
[Condition] v783 (AGT:n124, n507, THM:n734,n360)
→ **[Conclusion]NEG** (THM : n734,n360)
[Exception Condition] n360,n350,n742,n900
→[Exception Conclusion]NEG NEG (THM :n734, n360)

====================================

To compare the semantic content between logical representations is difficult. So, we use a logical representation only for the identification of easy questions. If the concepts (Kadokawa thesaurus numbers) of the predicates and arguments of a query appear in the corresponding article, then we consider the question is easy, for which we just count the negation level and return the answer. If there are multiple concepts for an argument (or predicate) in a query, we consider it is acceptable if any one concept appears in the corresponding article. In our training data, 47.85% of the data were assigned as easy questions, and 52.15% were assigned as non-easy questions.

The negation level (*neg_level()*) is computed as following: if negation(NEG) occurs an odd number of times, its negation level is 1. Otherwise if the negation(NEG) occurs an even number of times, including zero, its negation level is 0. If the *neg_level()* of the query condition is the same with that of the statute condition, and the *neg_level()* of the query conclusion is the same with that of the statute conclusion, then we consider the answer is 'yes' (the entailment is true), otherwise the answer is 'no' (the entailment is not true.)

The output of our logic-based system is also used below in an unsupervised learning model for assigning labels of condition and conclusion clusters for non-easy questions.

3.4 Step 2: Unsupervised machine learning

For the questions not confirmed as easy, we need to construct deeper representations. Fully general solutions are extremely difficult, if not impossible; for our first approximation to the non-easy cases, we have developed a method using unsupervised learning based on more detailed linguistic information. Since we do not know the impact each linguistic attribute has on our task, we use a machine learning algorithm that learns what information is relevant in the text to achieve our goal.

The types of features we use are as follows:

•**Negation feature** *(neg_level())*
•Syntactic representation features Considering condition, conclusion, and exception
•**Sematic representation features** Considering semantic role (argument, predicate)
•**Lexical semantic features** Having the same Kadokawa thesaurus concept code.

We use our learning method on linguistic features to confirm the following semantic entailment features:

$$\text{Feature 1}: if\,(concept(w_{main\_pred}), Query_{conclusion}) \cap (concept(w_{main\_pred}), Article_{conclusion})$$

$$\text{Feature 2}: If\ \exists_{i,j}(concept(w_{\arg i}), Query_{conclusion}) \cap (concept(w_{\arg j}), Article_{conclusion})$$

$$\text{Feature 3}: If\ \exists_{i,j}(concept(w_{pred_i}), Query_{condition}) \cap (concept(w_{pred_j}), Article_{condition})$$

$$\text{Feature 4}: If\ \exists_{i,j}(concept(w_{\arg i}), Query_{condition}) \cap (concept(w_{\arg j}), Article_{condition})$$

$$\text{Feature 5}: If\ neg\_level(Query_{condition}) = neg\_level(Article_{condition})$$

$$\text{Feature 6}: If\ neg\_level(Query_{conclusion}) = neg\_level(Article_{conclusion})$$

Table 4: Our Performances in Textual Entailment

Method	Accuracy
TF-IDF+TE	0.7179
LM+TE	0.6923

Table 5: Performance between our method vs. others for Textual Entailment

Method	Accuracy	Method	Accuracy
Our TF-IDF+TE	**0.7179**	KIS-YN-A	0.5385
iLis7	0.5641	KIS-YN-CM	0.5385
iLis9-1	0.5769	KIS-YN-CS	0.5897
iLis9-2	0.5385	KIS-YN-M	0.5769
JAISTNLP2-2a-1a	0.5128	KIS-YN-S	0.6538
JAISTNLP2-2a-1b	0.4744	NAIST1	0.6154
JAISTNLP2-2b-1a	0.4872	NAIST2	0.6538
JAISTNLP2-2b-1b	0.5000	NAIST3	0.4744
JNLP1-R	0.4359	NOR17	0.5385
JNLP1-RT	0.4872		

Feature 1 is intended to consider if the concepts of main predicates between a query conclusion and an article conclusion are the same. Most of the conclusion segment includes only one predicate (main predicate), so we just compare the meaning of the main predicates between a query and a law statute. We consider two concepts are the same, if their Kadokawa thesaurus numbers are the same. Features 2 and 4 check if there are overlapped concepts in the arguments between a query conclusion (condition) and its relevant article conclusion (condition). Feature 3 checks if there are overlapped concepts in predicates between a query condition and its relevant article condition. Features 5 and 6 check the negation levels between the query condition (conclusion), and corresponding article condition (conclusion).

The inputs for our unsupervised learning model are all the questions and corresponding articles. The outputs are two clusters of the questions. The yes/no outputs of easy questions (which have already been obtained) are used as a key for assigning yes/no label of each cluster. The cluster that includes a higher portion of yes of the easy questions is assigned the label 'yes', and the other cluster is assigned 'no'. For the non-easy questions, we determine their yes/no answers following their clustering labels. For the easy questions, we use results in Section 3.4 using *neg_level()*, regardless of the clustering labels of the questions, because the logic produces more accurate answers for easy questions than the clustering output. We use a simple K-means clustering algorithm with K=2 for unsupervised learning. We trained the K-means clustering algorithm using the Korean-translated training data, and then use the clusters to classify the unseen test data.

Table 6: Ablation analysis for our features

Method	Accuracy(%)
Our method using all steps	71.79
Without logical representation (1^{st} step)	61.54
Without machine learning (2^{nd} step)	44.87

Table 7: Error types of incorrectly answered questions

Error type	Accu. (%)	Error type	Accu. (%)
Specific example case	15.38	Semantic similarity error	34.62
Incorrect deletion of the most similar article sentence	14.10	More constraints in condition	20.51
Incorrect detection of condition, conclusion, and exceptional cases	10.25	Etc.	5.13

4 EXPERIMENTS

4.1 Experimental setup

In the general formulation of the textual entailment problem, given an input text sentence and a hypothesis sentence, the task is to make predictions about whether or not the hypothesis is entailed by the input sentence. We report the accuracy of our method in answering yes/no questions of legal bar exams by predicting whether the questions are entailed by the corresponding civil law articles.

There is a balanced positive-negative sample distribution in the training dataset (51.63% yes, and 48.37% no) of the COLIEE 2017 dataset, so we consider the baseline for true/false evaluation is the accuracy when returning always yes, which is 51.63%.

Our training data has 581 questions, with total 1044 civil law articles, and test data has 78 questions. We use an unsupervised learning method, since the data size is not big enough to separate it into training and test data.

4.2 Experimental results

Table 4 shows the experimental results for Phase 2 using the formal run data of COLIEE 2017. The formal run data size is 66 queries for Textual Entailment task.

Our performance is 71.79%, and it ranked highest in the COLIEE-2017 competition by 6.41%, compared to the 2nd ranked systems of KIS-YN-S and NAIST2 as shown in Table 5.

Table 6 shows the experimental results arising from adjusting some features in our method. When we used only one approach without combining logical form and machine learning, the performance was lower. When we did not use our logical representation, we considered only lexical words in the whole query/statute and negation level as features for the machine learning. When we did not use machine learning, we considered only the negation level.

4.3 Error analysis

From unsuccessful instances, we classified the error types as shown in Table 7. The biggest error arises, of course, from the semantic similarity error, and we believe our Kadokawa thesaurus is not sufficient to capture the required depth of semantic similarity. The second biggest error is because of complex constraints arising in statute conditions. As with the other error types, there are cases where a question is an example case of the corresponding article, and the corresponding article embeds another article. We also found cases that indicate the need to do more extensive temporal analysis.

In addition, because our logical representation does not embed modality information (such as "must", "may", etc.), it can also cause errors. In future work, we plan to extend the richness of the representation, as well as consider extending the data size (e.g., questions) in the legal domain.

5 RELATED WORK

A textual entailment method from Bdour et al. [2] provided the basis for a Yes/No Arabic Question Answering System. They used a kind of logical representation, which bridges the distinct representations of the functional structure obtained for questions and passages. Lien and Kouylekov [9] proposed semantic parsing for textual entailment. Nielsen et al. [14] extracted features from dependency paths, and combined them with word-alignment features in a mixture of an expert-based classifier. Zanzotto et al. [17] proposed a syntactic cross-pair similarity measure for RTE.

Harmeling [5] took a similar classification-based approach with transformation sequence features. Marsi et al. [12] described a system using dependency-based paraphrasing techniques. All these previous systems uniformly conclude that syntactic information is helpful in RTE. Like in this previous work, we also obtain syntactic information and construct semantic representation using that syntactic information.

There are many QA studies in the legal field. The first one is ResPubliQA 2009 [15]. It describes the first round of ResPubliQA, a Question Answering (QA) evaluation task over European legislation, proposed at the Cross Language Evaluation Forum (CLEF) 2009. The ResPubliQA 2009 exercise is aimed at retrieving answers to a set of 500 questions. The answer of a question is a paragraph of the test collection. The hypothetical user considered for this exercise is a person interested in making inquiries in the law domain, specifically on the European legislation. There is another system for QA of legal documents reported by Monroy et al. [13]. They used natural language techniques such as stemming, and resources such as manually or automatically constructed thesauri for improving question based document retrieval. In addition, there was a method based on syntactic tree matching [10], and a knowledge-based method using a variety of thesaurus and dictionaries [1]. As further research, we can enrich our knowledge base with deeper analysis of data, and add paraphrasing dictionary.

In previous Competitions on Legal Information Extraction/Entailment (COLIEE) 2014-2016 [8], the competition consists of three tasks: Legal Information retrieval (task 1), legal text entailment (task 2), and combination of the tasks 1 and 2 (task 3). Participants have applied a variety of machine learning skills and word features such as word embedding. However, there was no previous use of induction on logic-based representations.

In addition, there have been textual entailment challenges in the SemEval 2013-2014. Bjerva et al. [3] which showed good performance in recognizing textual entailment produced work as follows: (1) produce a formal semantic representation using a semantic parser Boxer for each sentence for a given sentence pair; (2) translate these semantic representations into first-order logic; and then (3) use off-the-shelf theorem provers and model builders to check whether the first sentence entails the second, or whether the sentences are contradictory. We simplify this process here, and construct simplified semantic representation using phrase chunking, exploit the structure of syntactic dependency trees, and semantic similarity using Kadokawa thesaurus mapping.

6 CONCLUSION

We have described our most recent implementation for the Competition on Legal Information Extraction/Entailment (COLIEE)-2017 Task.

For Textual entailment, we have proposed a method to answer yes/no questions from legal bar exams related to civil law. We used a two-step cascaded model using a logic-based representation and machine learning. In support, we constructed a negation dictionary. For the logical representation, we transformed a syntactic parse tree, augmenting Kadokawa thesaurus concept information. This method shows the best performance in Textual entailment task in the COLIEE-2017 competition.

ACKNOWLEDGEMENTS

This research was supported by the Alberta Machine Intelligence Institute (www.amii.ca). We are indebted to Ken Satoh of the National Institute for Informatics, who had the vision to create the COLIEE competition.

REFERENCES

[1] Arti Arya, Vishwanath Yaligar, Ramya D. Prabhu, Ramya Reddy, and Rohith Acharaya. A knowledge based approach for recognizing textual entailment for natural language inference using data mining. *International Journal on Computer Science and Engineering*, 02(06):2133–2140, 2010.

[2] Wafa N. Bdour and Natheer K. Gharaibeh. Development of yes/no arabic question answering system. *International Journal of Artificial Intelligence and*

Applications, 4(1):51–63. 2013.

[3] Johannes Bjerva, Johan Bos, Rob V.d. Goot, and Malvina Nissim. The meaning factory: Formal semantics for recognizing textual entailment and determining semantic similarity. *In Proceedings of SemEval*, pages 642–646, 2014.

[4] Susumu Ono, and Masando Hamanishi, New Synonym Dictionary. Kadokawa Shoten. Japan. 1981

[5] Stefan Harmeling. 2007. An extensible probabilistic transformation-based approach to the third recognizing textual entailment challenge. In *Proceedings of ACL PASCAL Workshop on Textual Entailment and Paraphrasing*, pages 137–142, 2007.

[6] K. Sparck Jones, A statistical interpretation of term specicity and its application in retrieval. In: Willett, P. (ed.) Document Retrieval Systems, pp. 132-142. Taylor Graham Publishing, London, UK, UK, 1988.

[7] Mi-Young Kim, Sin-Jae Kang, and Jong-Hyeok Lee. 2001. Resolving ambiguity in inter-chunk dependency parsing. In *Proceedings of 6th Natural Language Processing Pacific Rim Symposium*, pages 263–270, 2001.

[8] Mi-Young Kim, Randy Goebel, Yoshinobu Kano, and Ken Satoh. COLIEE-2016: Evaluation of the Competition on Legal Information Extraction and Entailment, In International Workshop on Juris-informatics (JURISIN 2016), 2016

[9] Elisabeth Lien and Milen Kouylekov. Semantic parsing for textual entailment. In *Proceedings of the International Conference on Parsing Technologies*, pages 40–49, 2015.

[10] Zhewei Mai, Yue Zhang, and Dong-Hong Ji. Recognizing text entailment via syntactic tree matchng. In *Proceedings of NTCIR-9 Workshop meeting*, pages 361–364, 2011.

[11] Christopher D. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. J. Bethard, and D. McClosky. *The Stanford CoreNLP Natural Language Processing Toolkit.* In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations, pp. 55-60, 2014.

[12] Erwin Marsi, Emiel Krahmer, and Wauter Bosma .Dependency-based paraphrasing for recognizing textual entailment. In *Proceedings of ACL PASCAL Workshop on Textual Entailment and Paraphrasing*, pages 83–88, 2007.

[13] Alfredo Monroy, Hiram Calvo, and Alexander Gelbukh. Nlp for shallow question answering of legal documents using graphs. Lecture Notes in Computer Science (from CICLING 2009), 5449:498–508, 2009.

[14] Rodney D. Nielsen, WayneWard, and James H. Martin. Toward depend-ency path based entailment. In *Proceedings of the second PASCAL Challenges Workshop on RTE*, pages 44–49, 2006.

[15] Anselmo Peas, Pamela Forner, Richard Sutcliffe, lvaro Rodrigo, Corina Forscu, Iaki Alegria, Danilo Giampiccolo, Nicolas Moreau, and Petya Osenova. Overview of respubliqa 2009: Question answering evaluation over euro-pean legislation. *Lecture Notes in Computer Science (from CLEF 2009 Workshop),* 6241:174–196, 2009.

[16] Jay M. Ponte and Bruce Croft. "A language modeling approach to information retrieval." In Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval, pp. 275-281. ACM, 1998.

[17] Fabio M. Zanzotto, Alessandro Moschitti, Marco Pennacchiotti, and Maria T. Pazienza. Learning textual entailment from examples. In *Proceedings of the second PASCAL Challenges Workshop on RTE*, 2006

[18] M-Y. Kim, Y. Xu, and R. Goebel. Alberta-KXG: Legal Question Answering Using Ranking SVM and Syntactic/Semantic Similarity. Eighth International Workshop on Jurisinformatics (JURISIN), 2014