Legal Question Answering Using Ranking SVM and Syntactic/Semantic Similarity

Conference Paper · August 2015

DOI: 10.1007/978-3-662-48119-6_18

3 authors, including:

Mi-Young Kim
University of Alberta
8 PUBLICATIONS **42** CITATIONS

SEE PROFILE

Randy Goebel
University of Alberta
206 PUBLICATIONS **2,808** CITATIONS

SEE PROFILE

New Frontiers in Artificial IntelligenceVolume 9067 of the series Lecture Notes in Computer Science pp 244-258

Legal Question Answering Using Ranking SVM and Syntactic/Semantic Similarity

Mi-Young Kim, Ying Xu, and Randy Goebel

Dept. of Computing Science, University of Alberta, Edmonton, Canada

{ miyoung2,yx2,rgoebel}@ualberta.ca

Abstract. We describe a legal question answering system which combines legal information retrieval and textual entailment. We have evaluated our system using the data from the first competition on legal information extraction/entailment (COLIEE) 2014. The competition focuses on two aspects of legal information processing related to answering yes/no questions from Japanese legal bar exams. The shared task consists of two phases: legal ad-hoc information retrieval and textual entailment. The first phase requires the identification of Japan civil law articles relevant to a legal bar exam query. We have implemented two unsupervised baseline models (tf-idf and Latent Dirichlet Allocation(LDA)-based Information Retrieval(IR)), and a supervised model, Ranking SVM, for the task. The features of the model are a set of words, and scores of an article based on the corresponding baseline models. The results show that the Ranking SVM model nearly doubles the Mean Average Precision compared with both baseline models. The second phase is to answer "Yes" or "No" to previously unseen queries, by comparing the meanings of queries with relevant articles. The features used for phase two are syntactic/semantic similarities and identification of negation/antonym relations. The results show that our method, combined with rule-based model and the unsupervised model, outperforms the SVM-based supervised model.

Keywords: legal text mining, question answering, recognizing textual entailment, information retrieval, ranking SVM, Latent Dirichlet Allocation (LDA)

1. Task Description

The Competition on Legal Information Extraction/Entailment (COLIEE) 2014 focuses on two aspects of legal information processing related to answering yes/no questions from legal bar exams: Legal document retrieval (phase 1) and Yes/No Question answering for legal queries (phase 2).

Phase 1 is an ad-hoc information retrieval (IR) task. The goal is to retrieve relevant Japan civil law articles that are related to a question in legal bar exams. Here **relevant** means, based on the articles, lawyers are able to infer the question's answer.

We approach this problem with two unsupervised models based on statistical information. One is the tf-idf model [1], i.e. term frequency-inverse document frequency. The relevance between a query and a document depends on their

intersection word set. The importance of words is measured with a function of term frequency and document frequency as parameters.

The other unsupervised model is a topic model-based IR system. The topic model we employ is the Latent Dirichlet Allocation (LDA) model [2]. The LDA model assumes that there is a set of latent topics for a corpus. A document can have several topics, and words in the document are assumed to be generated based on the topic distribution of such a topic model. The more similar two documents are, the more similar their topic distributions will be. While the tf-idf model considers only words that are both in the query and the document, the LDA based IR model also considers words that are in the query but not in the document. For example, if "seller" is only in the query and the document is about trading and contains words such as "buy," "customer," etc., then the probability that the document generates "seller" will be high.

In addition to implementing both models for the task, we also incorporate them into a Ranking SVM model [3]. That model is used to re-rank documents that are retrieved by the tf-idf model. The model's features include lexical words, and both tf-idf score and LDA model score of a document for a given query. The intuition is that, by including lexical words, the SVM model can re-assign weights to lexical words based on the training instances instead of solely on statistical information. By including both statistical models' scores, the SVM model can take advantage of both models.

Our experiments for phase 1 show that all features contribute to the improvement of IR results. The Ranking SVM model nearly doubles the Mean Average Precision (MAP) compared with both baseline models.

The goal of phase 2 is to construct Yes/No question answering systems for legal queries, by entailment from the relevant articles. The task investigates the performance of systems that answer "Y" or "N" to previously unseen queries, by comparing approximations to the semantics of queries and relevant articles.

Our system uses features that depend on the syntactic structure, the presence of negation, and the semantic similarities of the words. The combined model with rules and unsupervised learning model shows reasonable performance, which improves the baseline system, and outperforms an SVM-based supervised machine learning model.

2. Phase 1: Legal Information Retrieval

2.1 IR Models

In this section, we introduce our tf-idf model, our LDA-based IR model, and our Ranking SVM model. Queries and articles are all tokenized and parsed by the Stanford NLP tool. For the IR task, the similarity of a query and an article is based on the terms within them. Our terms can be a word or a dependency linked word bigram.

2.1.1 TF-IDF

One of our baseline models is a tf-idf model implemented in Lucene, an open source IR system[1].

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

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

The score *tf-idf(Q,A)* is a measure which computes the relevance between a query *Q* and an article *A*. First, for every term *t* in the 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*. After some normalization process in the Lucene package, we multiply *tf(t,A)* and *idf(t)*, and then we compute the sum of these multiplication scores for all terms *t* in the query *A*. This summation result is *tf-idf(Q,A)*. The bigger *tf-idf(Q,A)* is, the more relevant between the query *Q* and the article *A*. The real version has some normalized parameters in terms of an article's length to alleviate the functions biased towards long documents.

2.1.2 LDA-based IR

Our LDA-based IR model was first proposed in [4]. For more details about LDA model please refer to [2]. The score function is:

$$LDA(Q,A)=\prod_{t\in Q}P(t\mid A) \quad (2)$$

P(t|A) is the probability of the term *t* given the article *A*, specified as:

$$P(t\mid A)=\sum_{z\in K}P(t\mid z)\times P(z\mid A) \quad (3)$$

where *z* is a latent topic in topic set *K*, *P(t|z)* is the probability of the term *t* given topic *z*, and *P(z|A)* is the probability of the topic *z* given the article *A*. The two probabilities are from an LDA-model trained with the Mallet package[2].

2.1.3 Ranking SVM

The Ranking SVM model was proposed by [3]. The model ranks the documents based on user's click through data. Given the feature vector of a training instance, i.e. a retrieved article set given a query, denoted by $\Phi(Q,A_i)$, the model tries to find a ranking that satisfies constraints:

$$\Phi(Q,A_i)\rangle\Phi(Q,A_j) \quad (4)$$

where A_i is a relevant article for the query *Q*, while A_j is less relevant.

---

[1] Lucene can be downloaded from http://lucene.apache.org/core/.

[2] Mallet is a machine learning software package. It can be downloaded from http://mallet.cs.umass.edu/

We incorporate four types of features.

- Lexical words: the lemmatized normal form of surface structure of words in both the retrieved article and the query. In the conversion to the SVM's instance representation, this type is converted into binary features whose values are one or zero, i.e. if a word exists in the intersection word set or not.
- Dependency pairs: word pairs that are linked by a dependency link. The intuition is that, compared with the bag of words information, syntactic information should improve the capture of salient semantic content. Dependency parse features have been used in many NLP tasks, and improved IR performance [5]. This feature type is also converted into binary values.
- TF-IDF score (Section 2.1.1).
- LDA-based IR score (Section 2.1.2).

2.2 Experiments

The legal IR task has several sets of queries with the Japan civil law articles as documents (755 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 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 4 sets of queries for a dry run. The 4 sets of data include 179 queries, and 223 relevant articles (average 1.25 articles per query). We used the corresponding 4-fold leave-one-out cross validation evaluation. The metrics for measuring our IR models is Mean Average Precision (MAP):

$$\mathrm{MAP}(Q)=\frac{1}{|Q|}\sum_{q\in Q}\frac{1}{m}\sum_{k\in(1,m)}\mathrm{precision}(R_k) \tag{5}$$

where Q is the set of queries, and m is the number of retrieved articles. R_k is the set of ranked retrieval results from the top until the *k-th* article. In the following experiments, we set two values of m, which are 5 and 10, for all queries, corresponding to the column MAP@5 and MAP@10 in Table 1.

The following experiments compare different IR models on the legal IR task, verifying whether the ensemble SVM-Ranking model improves the IR performance.

The LDA model is trained on the whole Japan civil law dataset. The parameters such as number of training iterations and number of topics are set according to the 4-

fold cross validation IR performance. SVM's parameters are in a similar manner. Given the top 20 articles returned by the tf-idf model, the SVM model extracts features for every article and trains according to the order that correct articles are ranked higher than incorrect ones. The LDA values used as a feature in the SVM models are from the best LDA-based IR system. Table 1 presents the results of different models. The result shows that including dependency information slightly improves IR. The ranking SVM almost doubles the MAP. It does learn a better term weight for IR systems. Including the baseline tf-idf scores as features slightly improves the performance, and including LDA-based score further improves the performance.

Table 2 shows the IR result on test data. This result was obtained by using the model Id 6. We did not use the model Id 8 because of the relatively low performance gain after adding LDA information. The test data consists of 47 questions and 51 corresponding articles (average 1.09 relevant articles per question). As shown in Table 2, our system produced 61 relevant articles, and precision of 60.66%, recall of 72.55%, and F-measure of 66.07%. This performance was ranked No. 1 for the phase 1 of the competition.

Id	models	MAP@10	MAP@5
1	tf-idf with lemma	0.108	0.154
2	LDA-based	0.105	0.147
3	tf-idf with lemma and dependency pair	0.111	0.158
4	SVM-ranking with lemma	0.199	0.287
5	SVM-ranking with lemma and dependency pair	0.204	0.294
6	model 5 plus tf-idf score	0.211	0.306
7	model 5 plus LDA-based score	0.207	0.298
8	model 5 plus LDA-based score plus tf-idf score	**0.214**	**0.312**

Table 1. IR results on dry run data with different models.

Participant ID	Performance on Phase one	
Alberta-KXG	* The number of submitted articles: 61 * The number of correctly submitted articles: 37	
	Precision	0.6066
	Recall	0.7255
	F-measure	**0.6607**

Table 2. IR results on test data using the model ID. 6

2.3 Discussion

The baseline tf-idf model's performance confirms that legal case information retrieval is a difficult task. One problem is language variability, i.e., different

expressions convey the same meaning. For explanation purposes, consider the following example:

Question: In a demand for compensation based on delayed performance of loan claim, if person Y gives proof that the delayed performance is not based on the reasons attributable to him or herself, person Y shall be relieved of the liability.

Related Article: (Special Provisions for Monetary Debt)Article 419(1)The amount of the damages for failure to perform any obligation for the delivery of any money shall be determined with reference to the statutory interest rate; provided, however, that, in cases the agreed interest rate exceeds the statutory interest rate, the agreed interest rate shall prevail.(2)The obligee shall not be required to prove his/her damages with respect to the damages set forth in the preceding paragraph.(3)The obligor may not raise the defense of force majeure with respect to the damages referred to in paragraph 1.

There is no common content word between the question and the article. An expert can consider several alignments to help decide that the two paragraphs are related: "force majeure"' can be aligned to the negative of "reasons attributable to him or herself;" "agreed interest rate shall prevail" can be aligned to "person Y shall be relieved of the liability;" and "failure to perform" can be aligned to "delay;" etc.

From this example, we can see the difficulty of the task. It cannot solely depend on lexical information, but also take semantics into consideration. Any approach to address this problem will require at least a larger corpus from which to build improved semantic models.

3. Phase 2: Answering 'Yes'/'No' Questions based on Textual Entailment

Our system combines different kinds of syntactic and semantic information to predict textual entailment. We exploit syntactic/semantic similarity features, and negation. For negation and antonym patterns, we use M-Y.Kim et al.[6]'s approach.

3.1 Our System

3.1.1 Condition and Conclusion Detection

Previous Textual Entailment systems compute lexical or syntactic similarities between two whole sentences [7-10]. However, the sentences in this task are more complex; for example they often contain components like condition (premise), conclusion, and exceptional case description. Even when two sentences have similar word occurrences, if condition and conclusion are placed in a different order, or if a condition part matches an exceptional case, then the result of textual entailment should be 'no.' Therefore, we first need to identify condition, conclusion, and exceptional sentences in civil law articles, before computing similarity.

We also exploit keywords indicating a condition within dry run data. The keywords are as follows: 'in case(s) ," "if," "unless," "with respect to," "when," and comma. The sentence which explain exceptional cases also have keywords such as "provided," and "not apply," and those sentences typically appear in the last position in an article.

We can split the articles into three parts using these keywords. Condition and conclusion are in the first sentence, but an exceptional case is explained in the second sentence. There also exist articles that do not mention exceptional cases.

$$conclusion := segment_{last}(first\ sentence, keyword_{condition}),$$
$$condition := Concatenate(segment_i(first\ sentence, keyword_{condition})),$$
$$exception := second\ sentence\ which\ includes\ keyword_{exception}$$

Condition is the concatenation of segments that include the keywords for conditions in the first sentence. Conclusion is the last segment in the first sentence. An exceptional case is the second sentence that includes the keywords for exceptional cases.

For example,

<Civil Law Article 295-1>: 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.

(1) *Condition=> If a possessor of a Thing belonging to another person has a claim that has arisen with respect to that Thing,*
(2) *Conclusion=> he/she may retain that thing until that claim is satisfied.*
(3) *Exception=> Provided, however, that this shall not apply if such claim has not yet fallen due.*

3.1.2 Textual Entailment between a query sentence and several articles

Previous textual entailment systems compare two sentences and deduce a result [7-10]. However, in our task, a query sentence can have multiple relevant articles, and one article consists of several sentences. So, in order to propose the appropriate textual entailment result, we have to compare the meanings of one query sentence and multiple article sentences (one sentence vs. multiple articles).

Our approach assumes that there exists one most relevant article amongst relevant articles which can answer 'yes' or 'no' for a query sentence. For simplicity, we select the most relevant article and then try to deduce a textual entailment result by comparing between a query sentence and most relevant article (one sentence vs. one article). Here, we will describe how we choose the most relevant article with a query, from previously identified relevant articles:

most relevant article sentence :=

$$\arg\max_{article_{n,j} \in \text{articles relevant with } query_n} \{ overlap(condition_{article_{n,j}}, condition_{query_n}) + overlap(conclusion_{article_{n,j}}, conclusion_{query_n}) \}$$

The basic idea is that sentence $article_{n,j}$ is the most relevant article if it has maximum word overlap with the $query_n$ sentence. $condition_s$ is the condition part of the sentence s, and $conclusion_s$ is the conclusion of the sentence s. When we compute the word overlap, we separately compute the word overlap in condition parts and conclusion parts, and then use their sum.

For this approach to textual entailment, we have tried three methods: applying hand-constructed rules, creating a simple model with unsupervised learning, and then using an SVM-based supervised learning method.

3.1.3 Applying rules

if ($neg\_level(condition_{article\ n}) + neg\_level(conclusion_{article\ n})$
$= neg\_level(condition_{query\ n}) + neg\_level(conclusion_{query\ n})$),
$Answer_n := yes$,
otherwise, $Answer_n := no$,
where *neg_level() := 1 if negation and antonym occur odd number of times.*
neg_level() := 0 otherwise.

Figure 1. Answering rule for easy questions

Because our language domain is restricted for both the input questions and law articles, there are some questions that can be answered easily using only negation and antonym information. If the question and article share the same word as the root in each syntactic tree, we consider the question as easy, which means it can be answered using only negation/antonym detection. Here is an example:

Question : If person A sells owned land X to person B, but soon after, sells the same land X to person C then if the registration title is transferred to B, then person B can assert against C in the acquisition of ownership of land X.

Article 177 : Acquisitions of, losses of and changes in real rights concerning immovable properties may not be asserted against third parties, unless the same are registered pursuant to the applicable provisions of the Real Estate Registration Act and other laws regarding registration.

The conclusion and condition segments obtained from our system are as follows:

Conclusion of the question : then person B can assert against C in the acquisition of ownership of land X.

Condition of the question : : If person A sells owned land X to person B, but soon after, sells the same land X to person C then if the registration title is transferred to B,

*Conclusion of the article : Acquisitions of, losses of and changes in real rights concerning immovable properties may **not** be asserted against third parties,*
*Condition of the article : **unless** the same are registered pursuant to the applicable provisions of the Real Estate Registration Act and other laws regarding registration.*

In the above example, the conclusions in both the question and the article use the root word "assert" of the corresponding syntactic trees. Therefore, this example can be answered using only the confirming negation and antonym information. If the sum of the negation levels of a question is the same with that of the corresponding article, then we determine the answer is "yes," and otherwise "no."

The negation level is computed as following: if [negation + antonym] occurs an odd number of times in a condition (conclusion), its negation level is "1." Otherwise if the [negation + antonym] occurs an even number of times, its negation level is "0." In the above example, the negation level of the condition of the question is zero, and that of the conclusion of the question is also zero. The negation level of a condition of the article is one, and that of a conclusion of the article is also one. Since the sum of the negation levels of the question is the same with that of the corresponding article, we determine the answer of the question is "yes."

A more concise description for this rule is shown in Figure 1. In this Figure, $article_n$ is the most relevant article of the query $query_n$. The output of our rule-based system is also used below in an unsupervised learning model for assigning labels of condition (conclusion) clusters.

3.1.4 Unsupervised learning

We also try to construct a deeper representation for complex sentences. Fully general solutions are extremely difficult, if not impossible; for our first approximation for the non-easy cases, we have developed a method using unsupervised learning with more detailed linguistic information. Since we do not know the impact each linguistic attribute has on our task, we first run 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:

Word matching Having the same lemma.
Tree structure features Considering only the dependents of a root.
Lexical semantic features Having the same Kadokawa [11] thesaurus concept code.

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

$$\text{Feature 1}: if\ w_{root}(condition_{query_n}) = w_{root}(condition_{article_n})$$
$$\text{Feature 2}: if\ w_{root}(conclusion_{query_n}) = w_{root}(conclusion_{article_n})$$
$$\text{Feature 3}: if\ W_{dep}(conclusion_{query_n}) \cap W_{dep}(conclusion_{article_n}) \neq \phi$$
$$\text{Feature 4}: if\ c_{root}(condition_{query_n}) = c_{root}(condition_{article_n})$$
$$\text{Feature 5}: if\ c_{root}(conclusion_{query_n}) = c_{root}(conclusion_{article_n})$$
$$\text{Feature 6}: if\ neg\_level(condition_{query_n}) = neg\_level(condition_{article_n})$$
$$\text{Feature 7}: if\ neg\_level(conclusion_{query_n}) = neg\_level(conclusion_{article_n})$$

In the features above, $article_n$ is the most relevant article of the query $query_n$. $w_{root}(s)$ means the root word in the syntactic tree of the sentence s, and $W_{dep}(s)$ is the set of all the dependents of the root word in the syntactic tree of the sentence s. $c_{root}(s)$ is the Kadokawa concept code of the root word in the syntactic tree of the sentence s. For Features 1 and 2, we check if the root word in the syntactic tree of the $condition_{query\_n}(conclusion_{query\_n})$ is the same with that of the $condition_{article\_n}$ $(conclusion_{article\_n})$. For Feature 3, we examine if one of the dependents of the root word in the syntactic tree of the $conclusion_{query\_n}$ is also a dependent of the root word in the syntactic tree of the $conclusion_{article\_n}$. For Features 4 and 5, we check if the Kadokawa concept code of the root word in the syntactic tree of the $condition_{query\_n}$ $(conclusion_{query\_n})$ is the same with that of the $condition_{article\_n}(conclusion_{article\_n})$. For Features 6 and 7, we compare $neg\_level()$ between $condition_{query\_n}$ $(conclusion_{query\_n})$ and $condition_{article\_n}(conclusion_{article\_n})$. Features 1, 2, 3 consider both lexical and syntactic information, and Features 4 and 5 consider semantic information. Features 6 and 7 incorporate negation and antonym information. Features 1 and 2 are used to check if conditions (conclusions) of a question and corresponding article share the same root word in the syntactic tree. Feature 3 is used to determine if each dependent of a root in the conclusion of a question appears in the article. We heuristically limit the number of dependents as those three nearest to the root. Features 4 and 5 confirm if the root words of conditions (conclusions) of the question and corresponding article share the same concept code. We use some morphological and syntactic analysis to extract lemma and dependency information. Details of the morphological and syntactic analyzer are given in Section 3.2.

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 based on rules described in Section 2.2.3 are used as a key for assigning a yes/no label of each cluster. The cluster which includes higher portion of "yes" of the easy questions is assigned the label "yes," and the other cluster is assigned "no." We determine their yes/no answers using their clustering labels.

3.1.5 Supervised learning with SVM

We compare our method with SVM, as a kind of supervised learning model. Using the SVM tool included in the Weka [12] software, we performed cross-validation for the 179 questions using 7 features explained in Section 2.2.4. We used a linear kernel SVM because it is popular for real-time applications as they enjoy both faster training and classification speeds, with significantly less memory requirements than non-linear kernels because of the compact representation of the decision function.

3.2 Experimental setup for Phase 2

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 can be entailed by the corresponding civil law articles.

There is a balanced positive-negative sample distribution in the dataset (55.87% yes, and 44.13% no) for dry run, so we consider the baseline for true/false evaluation is the accuracy when returning always "yes," which is 55.87%. Our data for dry run has 179 questions, with total 1044 civil law articles.

The original examinations are provided in Japanese and English, and our initial implementation used a Korean translation, provided by the Excite translation tool (http://excite.translation.jp/world/). The reason that we chose Korean is that we have a team member whose native language is Korean, and the characteristics of Korean and Japanese language are similar. In addition, the translation quality between two languages ensures relatively stable performance. Because our study team includes a Korean researcher, we can easily analyze the errors and intermediate rules in Korean. We used a Korean morphological analyzer and dependency parser [13], which extracts enriched information including the use of the Kadokawa thesaurus for lexical semantic information. We use a simple unsupervised learning method, since the data size is not big enough to separate it into training and test data.

3.3 Experimental results

Our method	Accuracy (%)
Baseline	55.87
Rule-based model	53.77
Unsupervised learning (K-means)	60.85
Cross-validation with Supervised learning (SVM)	59.43
Rule-based model for easy questions + unsupervised learning for non-easy questions (combined)	61.96

Table 3. Experimental results on dry run data for phase 2

Evaluation of question answering systems is in general almost as complex as the construction of the question-answering itself. So one must make the choice to consider several features of QA systems in the evaluation process, e.g., query language difficulty, content language difficulty, question difficulty, usability, accuracy, confidence, speed and breadth of domain [14].

Table 3 shows our results on the dry run data. A rule-based model showed accuracy of 53.77%. We also use a K-means clustering algorithm with K=2 for unsupervised learning for the rest of the questions, and it showed accuracy of 60.85%. The overall performance when combining the use of rules and unsupervised learning showed 61.96% of accuracy which outperformed unsupervised learning for all questions, and even SVM (59.43%), the supervised learning model we use with a linear kernel. According to p-value measures (p=0.01) between the baseline and the combined model in the true/false determination, the combined model with rule-based model for easy questions and unsupervised learning model for non-easy questions significantly outperformed the baseline. Since previous methods use supervised learning with syntactic and lexical information, we consider the supervised learning experiment with SVM in Table 2 approximately represents the performance of previous methods.

Table 4 shows our performance of Phase 2 on test data. The test data consists of 41 questions, where 20 questions were 'Yes', and 21 questions were 'No'. The accuracy on test data is 63.41%, and it ranked No.1 on the COLIEE competition.

Our model	Accuracy on phase 2 (%)
Rule-based model for easy questions + unsupervised learning for non-easy questions (combined)	63.41

Table 4. Experimental results on test data for phase 2

We also evaluated the performance combining our two systems for phase 1 and phase 2. Table 5 shows the accuracies of yes/no answers of two different systems. The first system is our textual entailment system, which receives the legal bar exam queries and relevant articles as input, and produces yes/no answers for the input queries by entailment between queries and relevant articles retrieved by legal experts. The second system is the combined system of legal IR and textual entailment. The combined system receives only legal bar exam queries as input. Then, our legal IR system retrieves relevant articles for the queries, and our textual entailment system determines “yes” or “no” for each query through textual entailment. It is natural that the combined system shows poorer performance than that using only textual entailment system: this is because, in the combined system, the relevant articles were retrieved by our legal IR system, while in the textual entailment system, the relevant articles were confirmed by human experts.

Our system	Accuracy of 'Yes/No' answers (%)
Textual entailment system. (Input: legal bar exam queries and relevant articles)	67.39
Combined system of legal IR and textual entailment (Input : legal bar exam queries)	60.87

Table 5. Performance after combining two systems for phase 1 and phase 2

Error type	Accuracy (%)	Error type	Accu.(%)
Negation/Antonym	9.68	Paraphrasing	54.84
Exceptional case	12.90	Constraints in condition	9.68
Condition, conclusion mismatch	6.45	Other errors	6.45

Table 6. Error types in the result of the textual entailment system

We used the test data which was released for phase 1 excluding the last query (we've found that the last query was omitted in the translation process), and the accuracy of 'Yes/No' answers for our textual entailment system was 67.39% as shown in Table 5. The combined system showed 6.52% decrease of accuracy in answering "Yes/No." But we found some questions which were correctly answered, even though the retrieved relevant articles were incorrect. There is an obvious need for deeper and detailed analysis, which will require at least the integration of information extraction techniques to identify and exploit legal relationships.

3.4 Discussion

From unsuccessful instances, we classified the error types as shown in Table 6. We can see that the challenge of paraphrasing causes most of the errors of our system. As just mentioned, the broad challenges of accurate Question Answering are contingent on legal relationship identification and exploitation. The development of that knowledge could be addressed by exploiting expert knowledge and much larger corpora, in companion with existing automatic information extraction methods. One of the most obvious places to focus on information extraction is the need to do more extensive temporal analysis.

Table 6 also shows that the error rate attributed to problems of negation/antonym was 9.68%. This error mostly arose from the weakness of our antonym dictionary, as there were no errors of misinterpreted negations in the translation process.

The rate of the incorrectly translated sentences was 15.22% in the test data. Because the original sentences are relatively clear and the syntactic characteristics of Japanese and Korean are so similar, there were only a 4.35% error rate on syntactic structures in the translated sentences. Most of the errors resulted from the unnatural translation of a word in a sentence. It will be interesting to compare our performance using Korean-translated sentences with that using original sentences, in which case we would expect that using original sentences will show better performance.

In the legal bar exam, the human passing score is usually around 63% and our system achieved 60.87%. We think our performance in the first COLIEE competition is promising. For more accurate comparison with the human scores, we need to test the whole legal bar exam of a specific year including all difficult cases (e.g., a query identifies a legal article that refers to another legal article).

4 Related work

SemEval 2014 Task 1 [15] evaluates system predictions of semantic relatedness (SR) and textual entailment (TE) relations on sentence pairs from the SICK dataset, which consist of 750 images and two descriptions for each image. The top ranked system (UIUC) [16] in this task uses distributional constituent similarity and denotational constituent similarity features. Their method of comparing constituents between the whole two sentences are not useful in our task, because our task consists of one legal query and multiple corresponding article sentences.

There was a textual entailment method from W. Bdour et al. [17] which 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. This method is not appropriate for our task. If a false question sentence is constructed by replacing named entities with terms of different meaning in the legal article, a logic representation can be helpful. However, false questions are not simply constructed by substituting specific named entities, and any logical representation can make the problem more complex. Kouylekov and Magnini [18] experimented with various cost functions and found a combination scheme to work the best for RTE. Vanderwende et al. [19] used syntactic heuristic matching rules with a lexical-similarity back-off model. Nielsen et al. [20] extracted features from dependency paths, and combined them with word-alignment features in a mixture of an expert-based classifier. Zanzotto et al. [21] proposed a syntactic cross-pair similarity measure for RTE. Harmeling [22] took a similar classification-based approach with transformation sequence features. Marsi et al. [23] described a system using dependency-based paraphrasing techniques. All previous systems uniformly conclude that syntactic information is helpful in RTE: we also use syntactic information combined with lexical semantic information. As further research, we can enrich our knowledge base with deeper analysis of data, and add paraphrasing dictionary getting help from experts.

5 Conclusion

We have described our implementation for the Competition on Legal Information Extraction/Entailment (COLIEE) 2014 Task.

For phase 1, legal information retrieval, we implemented a Ranking-SVM model for the legal information retrieval task. By incorporating features such as lexical words, dependency links, tf-idf score, and LDA-based IR score, our model doubles the mean average precision.

For phase 2, we have proposed a method to answer yes/no questions from legal bar exams related to civil law. We used the knowledge base of M-Y. Kim et al.[6] by analyzing negation patterns and antonyms in the civil law articles. To make the alignment easy, we first segment questions and articles into condition, conclusion, and exception. We then extract deep linguistic features with lexical, syntactic information based on morphological analysis and dependency trees, and lexical semantic information using the Kadokawa thesaurus. Our method uses a hybrid model that combines a rule-based model for easy questions and unsupervised learning model for

non-easy questions. This achieved quite encouraging results in both true and false determination. We also show the performance combining the models for phase 1 and phase 2. To improve our approach in future work, we need to create deeper representations (e.g., to deal with paraphrase), and analyze the temporal aspects of legal sentences. In addition, we will complement our knowledge base with paraphrasing dictionary with the help of experts.

Acknowledgements

This research was supported by the Alberta Innovates Centre for Machine Learning (AICML) and the iCORE division of Alberta Innovates Technology Futures.

References

1. 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
2. D.M. Blei, A.Y. Ng, M.I. Jordan, Latent dirichlet allocation. J. Mach. Learn. Res .3, 993-1022, March 2003
3. T. Joachims, Optimizing search engines using clickthrough data. In: Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 133-142. KDD '02, ACM, New York, NY, USA, 2002
4. X. Wei, W.B. Croft, Lda-based document models for ad-hoc retrieval. In: Proceeding of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 178-185. SIGIR '06, ACM, New York, NY, USA, 2006
5. K.T. Maxwell, J. Oberlander, W.B. Croft. Feature-based selection of dependency paths in ad hoc information retrieval. In: Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 507-516. Association for Computational Linguistics, Sofia, Bulgaria, August 2013
6. M-Y. Kim, Y. Xu, R. Goebel, K. Satoh, “Answering Yes/No Questions in Legal Bar Exams”, JURISIN 2013
7. V. Jikoun and M. de Rijke. Recognizing textual entailment using lexical similarity. In Proceedings of the PASCAL Challenges Workshop on RTE, 2005
8. B. MacCartney, T. Grenager, M.-C. de Marneffe, D. Cer, and C. D. Manning. Learning to recognize features of valid textual entailments. In Proceedings of HLT-NAACL, 2006
9. R. Sno, L. Vanderwende, and A. Menezes. Effectively using syntax for recognizing false entailment. In Proceedings of HLT-NAACL, 2006
10. A. Lai, J. Hockenmaier, “Illinois-LH: A Denotational and Distributional Approach to Semantics”, In Proceedings of SemEval 2014: International Workshop on Semantic Evaluation. 2014
11. S. Ohno and M. Hamanishi, New Synonym Dictionary. Kadokawa Shoten, Tokyo, 1981
12. M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, I. H. Witten, The WEKA Data Mining Software: An Update; SIGKDD Explorations, Volume 11, Issue 1. 2009
13. M-Y. Kim, S-J. Kang and J-H. Lee, Resolving Ambiguity in Inter-chunk Dependency Parsing, Proc. of 6th Natural Language Processing Pacific Rim Symposium, pp. 263-270, 2001

14. M. Walas, How to answer yes/no spatial questions using qualitative reasoning?, Proc. of the International Conference on Computational Linguistics and Intelligent Text Processing, pp. 330-341, 2012
15. M. Marelli, L. Bentivogli, M. Baroni, R. Bernardi, S. Menini, and R. Zamparelli, "SemEval-2014 task 1: Evaluation of compositional distributional semantic models on full sentences through semantic relatedness and textual entailment", In Proceedings of SemEval 2014: International Workshop on Semantic Evaluation. 2014
16. A. Lai, J. Hockenmaier, "Illinois-LH: A Denotational and Distributional Approach to Semantics", In Proceedings of SemEval 2014: International Workshop on Semantic Evaluation. 2014
17. W. N. Bdour, and N.K. Gharaibeh, Development of Yes/No Arabic Question Answering System, International Journal of Artificial Intelligence and Applications, Vol.4, No.1 (51-63), 2013
18. M. Kouylekov, and B. Magnini. Tree edit distance for recognizing textual entailment: Estimating the cost of insertion. In Proceedings of the second PASCAL Challenges Workshop on RTE, 2006
19. L. Vanderwende, A. Menezes, and R. Snow. Microsoft research at rte-2: Syntactic contributions in the enta ilment task: an implementation. In Proceedings of the second PASCAL Challenges Workshop on RTE .2006
20. R. D. Nielsen, W. Ward, and J. H. Martin. Toward dependency path based entailment. In Proceedings of the second PASCAL Challenges Workshop on RTE, 2006
21. F. M. Zanzotto, A. Moschitti, M. Pennacchiotti, and M.T. Pazienza. Learning textual entailment from examples. In Proceedings of the second PASCAL Challenges Workshop on RTE, 2006
22. S. Harmeling, An extensible probabilistic transformation-based approach to the third recognizing textual entailment challenge. In Proceedings of ACL PASCAL Workshop on Textual Entailment and Paraphrasing, 2007
23. E. Marsi, E. Krahmer, and W. Bosma. Dependency-based paraphrasing for recognizing textual entailment. In Proceedings of ACL PASCAL Workshop on Textual Entailment and Paraphrasing, 2007