EPiC Series in Computing

Volume 47, 2017, Pages 79–89

COLIEE 2017. Fourth Competition on Legal Information Extraction/Entailment

Legal Question Answering System using Neural Attention

Ayaka Morimoto[1], Daiki Kubo[2], Motoki Sato[3], Hiroyuki Shindo[4], and Yuji Matsumoto[5]

[1] Graduate School of Information Science Nara Institute of Science and Technology, 8916-5 Takayama, Ikoma, Nara, 630-0192, Japan
morimoto.ayaka.lw1@is.naist.jp

[2] Graduate School of Information Science Nara Institute of Science and Technology, 8916-5 Takayama, Ikoma, Nara, 630-0192, Japan
kubo.daiki.kz7@is.naist.jp

[3] Graduate School of Information Science Nara Institute of Science and Technology, 8916-5 Takayama, Ikoma, Nara, 630-0192, Japan
sato.motoki.sa7.lw1@is.naist.jp

[4] Graduate School of Information Science Nara Institute of Science and Technology, 8916-5 Takayama, Ikoma, Nara, 630-0192, Japan
shindo@is.naist.jp

[5] Graduate School of Information Science Nara Institute of Science and Technology, 8916-5 Takayama, Ikoma, Nara, 630-0192, Japan
matsu@is.naist.jp

Abstract

This year's COLIEE has two tasks called phases 1 and 2. The phase 1 needs to find the relevant article given a query t2, and the phase 2 needs to answer whether the given query t2 is yes or no according to Japan civil law articles.

This paper presents our proposals for the phase 2 task. Two methods are presented. The first goes along the standard method taken by many authors, such that the relevant article t1 is selected by the similarity to the query t2 at the requirement (condition) and the effect (conclusion) descriptions of the articles. The second is our new proposal, in which Neural Networks with attention mechanism are applied to all the civil law articles in deciding the truthness of the query t2. This method takes into account all the articles by properly calculating their weighted sum.

1 Introduction

COLIEE is an annual competition for legal Information Extraction (IE) and Retrieval. COLIEE 2017 forcuses on two tasks, legal ad-hoc Information Extraction (IE) (Phase 1) and Textual Entailment (TE) (Phase 2).

While the Phase 1 can be regarded as the preprocessing for Phase 2, selecting inappropriate article(s) in Phase 1 may cause a bad effect on the Phase 2 task. We present two approaches to Phase 2, one is a standard method that selects the relevant article t1 for a legal bar exam query t2 and decides if t2 is true or not based on the selected article. In this method we tried

K.Satoh, M.-Y.Kim, Y.Kano, R.Goebel and T.Oliveira (eds.), COLIEE 2017 (EPiC Series in Computing, vol. 47), pp. 79–89

several methods to define word representations to estimate the similarity between articles. The second method, which is our new proposal, does not select the relevant article(s) for t2 but to makes use of Neural Networks with attention mechanism to utilize the information of all civil law articles to decide the correctness of the query t2.

2 Related Work

The major approaches taken by previous research were based on a two step method, to select the most relevant article t1 from the civil law articles as the evidence, then to apply a textual entailment method to decide if the query t2 is entailed by t1. The ensemble method proposed by [2] makes optimization of several weight parameters such as word overlap and tf-idf for selecting the relevant article t1 and applies a voting method with multiple classifiers. A heuristic method is proposed by [8], in which t1 is not explicitly selected. First, they make a one-to-one pair of the subject and the sentence end predicate using the result of a morphological analyzer and a case structure analyzer for each sentence. If the target phrase has no subject to be extracted, they only extract the sentence end predicate. Second, they simplify the extracted information to obtain better abstraction that helps to decide Yes/No. The method proposed by [3] uses tf-idf and Ranking SVM to select t1 and estimates the similarity between t1 and t2 based on the features of paraphrases and word embeddings coupled with condition/conclusion/exception analysis.

3 Preliminaries

This section introduces the base methods used in our two approaches, which are described in the following sections.

3.1 Word Representation

We used the idea of word2vec [7] for measuring the similarity between words and expressions. We tested several definitions for word segmentation for generating several word2vec models. As for the data for learning word embeddings, we used the judgment documents put on the web site of the Japanese Supreme Judicial Court[1], which contains 58,808 judgments (4M sentences).

We tested two sizes of word vectors, 50 and 200, when we use word2vec.

Definition of Words

Since Japanese is non-segmented language where there are no explicit word boundaries in written texts, we need to define proper units for word segmentation. The simplest method is to apply an existing word segmentation and POS tagging tool. For this approach, we used MeCab[2][4] and applied word2vec to the segmented documents. Two other segmentation criteria are also applied, one to attach suffixes to their preceding matrix phrases and the other to further attach functional expressions based on and extended from the dictionary of Japanese functional expressions[6]. Functional expressions in our case mainly mean multi-word expressions that work as postpositions or auxiliary verbs as a whole. Table 3.1 shows some examples of functional expressions. Those three segmentation criteria are simultaneously used to learn the embeddings of all possible word segmentations.

[1] http://www.courts.go.jp/app/hanrei_jp/search1
[2] http://taku910.github.io/mecab/

Functional expressions	English translation
(*ni-tui-te*)	about
(*to-ha-ie*)	however
(*koto-ga-dekiru*)	can
(*to-iu*)	that(complementizer)
(*sai-ni*)	when

Table 1: Examples of Functional Expressions

4 Similarity-based Method

The first method we applied is based on similarity between civil law articles and the given query at the requirement and the effect descriptions. The requirement and effect mean the condition and conclusion parts of Japan civil law articles and the exam queries. We use the surface clue expressions defined by [9] for identifying the requirement and effect parts of legal sentences.

4.1 Overview of the Method

The list of clue expressions that separate the requirement and effect parts of sentences in the law articles and the exam queries is shown in Table 4.2. In the current experiments we did not use the exceptional descriptions[3] often found in law articles.

The overall process is performed as the following. The details of each process is described in the subsequent subsections.

1. Extraction of the requirement and effect parts in law articles and query t2.
   - Clue expressions are used to identify requirement and effect parts.
2. Calculation of the similrity between law articles and t2 at both requirement and effect parts.
   - Previously learned word embeddings (using judgment documents) are used.
     - Attachment of suffixes and functional expressions is also considered.
   - Definition of similarity
     - word mover's distance is used.
   - Negative expressions are taken into consideration for selected articles, which flips the truth-value.

4.2 Extraction of requirement and effect parts

Japanese law adopts the Pandekten system. Therefore, the legal provisions contain the legal effect to be stipulated together with the legal requirements that constitute that condition. We used the method proposed by [9] to extract the legal requirement and effect parts from an article. First, we divide the sentence into parts separated by commas. Next, we extract the requirement part by the clue expressions shown in Table 4.2. The part matching a clue expression is regarded

[3]Typical law article sentences consist of condition and conclusion parts, possibly followed by one or more exceptional descriptions. We disregarded all the exceptional description in the current experiments.

Semantic classification	Normality	Surface patterns
Condition provision	strong	tokiha, tokiniha, saisiteha, atatteha, baaiha, baainiha, baainioiteha, baainohokaha, kagiriha, oiteha, ueha, nosaiha, unitsuiteha, runitsuiteha, reba, naraba
	weak	toki, tokini, saisite, atatte, baai, baaini, baainioite, baainohoka, kagiri, saisi, oite, ue, nosai, unitsuite, runitsuite
Status addition	weak	tokimo, tokinimo, saisitemo, atattemo, atattemo, baaimo, baainimo, baainioitemo, baaideattemo, baainohokanimo, oitemo, uemo, nosaimo, unitsuitemo, runitsuitemo
Separate judgment	strong	desadamerutokoroniyori, "number"niyori
Situation dependent	weak	jijouniyotte, jijouniyori
Means provision	weak	niyotte, niyori, motte, yotte
Objective provision	weak	you, youni, mokutekide, "lemma"utame
Applicable only	strong	taisiteha, tuiteha, kanshiteha
	weak	gendonioite, gendotoshite
Application addition	weak	taishitemo, tsuitemo, kanshitemo
Contents provision	weak	tsuite, kanshi, kanshite
Time provision	strong	madeha, madeniha, inainiha, inaiha, maeha, maeniha, atoha, atoniha, madenoaidaha, kanniha, kanha, chuuha, kikannaiha, tokiha, tokiniha, atonioiteha, maenioiteha, chuunioiteha
	weak	made, madeni, inaini, mae, maeni, ato, atoni, madenoaidani, aidani, kan, chu, kikannai, toki, tokini, atonioite, maenioite, chunioite, chitainaku
Time addition	weak	mademo, madenimo, inainimo, inaimo, atomo, atonimo, madenoaidamo, aidanimo, chumo, kikannaimo
Location regulation	strong	nainioiteha, shonioiteha
	weak	nainioite, shonioite
Location addition	strong	nainioitemo, shonioitemo
Application judgment	strong	mune
Cause specification		node
Applicable limit	weak	nouchi
Heterogeneous application	weak	monotoshite
Principle provision	weak	nishitagatte, nishitagai, nimotoduki, nimotoduite, niouji, nioujite

Table 2: Surface clue expressions for detecting requirement parts

as the requirement part, and the remaining part is regarded as the effect part. When a sentence includes two or more requirement parts, the effect part positioned right after the requirement parts is associated to each of the requirement parts to construct requirement-effect pairs.

4.3 Distance calculation

We calculate the distance between t2 and each article and extract the article t1 with the smallest distance from t2. We define the distance between sentences as the sum of "the distance between the requirement part of t2 and the requirement part of the article(t1)" and "the distance between the effect part of t2 and the effect part of the article(t1)". If the distance between t2

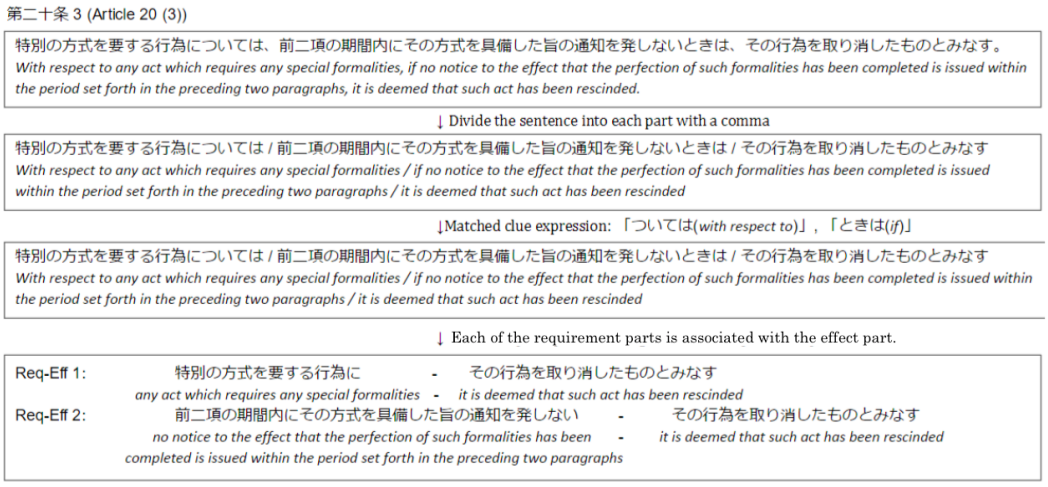

Figure 1: Process of extracting requirement-effect pairs

and the article extracted as t1 is equal to or less than a predefined threshold value, the system output is Y, and if it exceeds the threshold value, the output is N. We tested for threshold values, 25, 30, 35 and 40, and selected 35 for the experiments. The decision is made by a preliminary experiment using the H27 data (the previous year's data) as the development data. We used Word Mover's Distance as the distance measure. Word Mover's Distance is a method to calculate the distance between documents proposed by [5]. The calculation of the distance between documents is defined by the correspondence (association) or words in the compared documents. This method is applied to the calculation of the distance between sentences in our framework. The cost of corresponding words between the sentences is calculated by the distance of the distributed representation vectors, and the distance between sentences is calculated by the sum of the costs of corresponding words between the sentences.

4.3.1 Consideration of negation at sentence end

If the output of the system is Y, we consider whether the sentence end predicate[4] of either the requirement or effect part has a negative expression or not. If a negative expression is found, the answer is switched. If the article or t2 cannot be divided into the requirement and effect parts, the end predicate of the whole sentence is checked. Figure 2 shows how a negative expression is processed.

5 Attention Model

This section explains our attention model, which is based on Neural Networks with the attention mechanism[1]. We made an approach based on the idea of Memory Networks[10]. We learn vector representations for words in advance, and for given Japan civil law articles and the bar exam query t2, we calculate vector representations of articles and t2 as follows.

[4]Since Japanese is a head-final language, the main verb of a sentence always comes at the end of the sentence. Negative sentence usually have a negative auxiliary verb at the end.

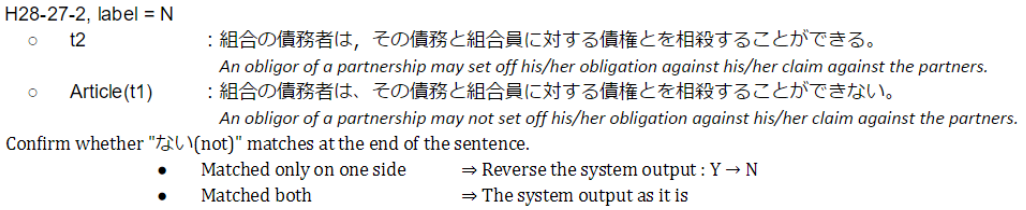

Figure 2: Process of consideration of negation

$$\vec{civilvec}_j = \sum_i civil\_idf_i \times \vec{x_i}$$

where $civilvec_j$ is the vector representation of the j-th article, which is a weighted sum of word representations $\vec{x_i}$ in the article. $civilvec$, representations of civil law articles, is a matrix with $civilvec_j$ as the j-th column. $civil\_idf$ is the idf value of the word x_i in the article[5], and $\vec{x}$ is the vector representation by word2vec learned with 60,000 jugement sentences. Those are the same representations used in the method in the previous section.

The vector for t2 is calculated in the same way as follows:

$$\vec{t2vec} = \sum_i t2\_idf_i \times \vec{x_i}$$

where $t2vec$ is the sum of word vectors in t2 weighted by their idf value. $t2\_idf$ is the idf value of the word x_i in t2, and $\vec{x}$ is the vector learned by word2vec, the same as those used for $civilvec$.

The calculation of attention is done by the inner product of the transpose of $civilvec$ and $t2vec$ defined as follows:

$$attention_j = \vec{civilvec_j}^{\,T} \cdot \vec{t2vec}$$

The size of the obtained attention vector is the number of articles. By taking the element-wise product of this attention and $civilvec$, we obtain the importance of articles, which is realized as the weighted sum of the articles, $attentioncivil$.

We then concatenate $attentioncivil$ and $t2vec$, and make it as the input of a multi-layer perceptron (MLP). We use three layer MLP, in which the number of units in the hidden layer is 50, and the size of the output unit for Y/N is 2. We used H18 to H26 for training and H27 for the development set.

Figure 3 shows the overall configuration of our Attention model.

In the MLP model, we tested several combinations of the parameters, such as the size of word2vec vectors, word2vec model themselves, activation functions and optimizers. Table 5 summarizes those options.

6 Experiments

The settings and the results of experiments are shown in this section.

[5]The idf values are calculated with all the civil law articles.

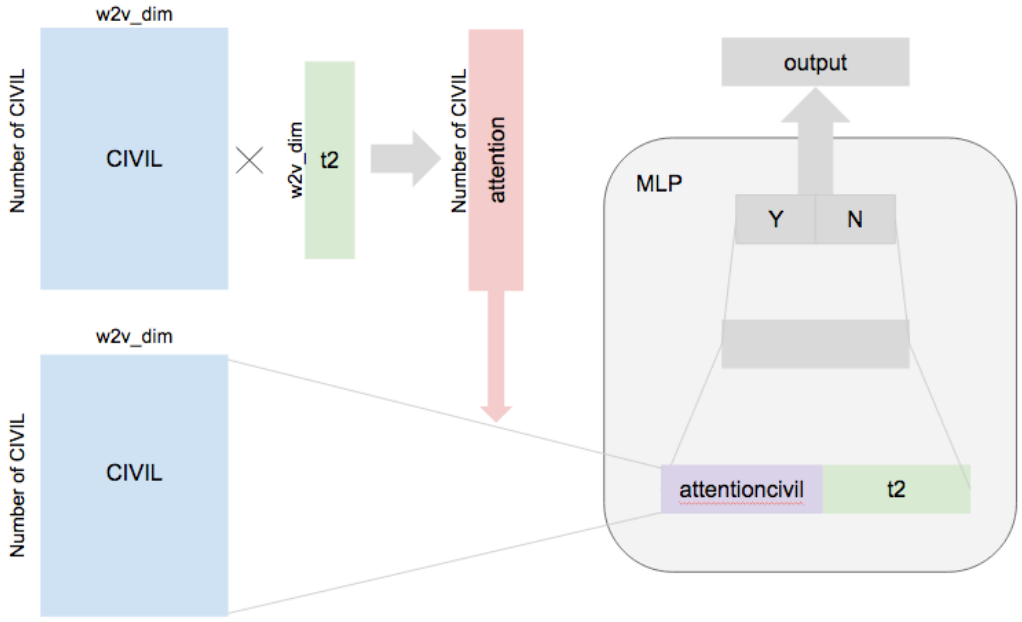

Figure 3: Image of Attention Model

word2vec(dimensions)	50, 200
word2vec(models)	traditional word units, Suffix attached, Suffix and functional expression attached
activation functions	relu, sigmoid, tanh
optimisers	AdaDelta, Adam, MomentumSGD(lr=0.01), SGD

Table 3: Parameter settings

6.1 Experimental setup

For the attention models, We will present the two submitted systems, NAIST1 and NAIST2, and the system that gave the best performance on the test data. We tested two sizes for word2vec, 50 and 200 dimensions. We hereafter report only the results with 200 dimensions since this setting achieved better results.

- NAIST1: sigmoid function for the activation function, and AdaDelta for the optimizer. Suffix and funcntional expressions are not considered.
- NAIST2: tanh for the activation function, and Adam for the optimizer. Suffix is considered, not functional expressions.
- sigMoMWE: sigmoid for the activation function, and MomentumSGD for the optimizer. Both suffix and functional expressions are considered.

	H18	H19	H20	H21	H22	H23
Separation of Requirement/Effect	0.37	0.50	0.49	0.57	0.55	0.63
+ negation	0.43	0.62	0.53	0.73	0.66	0.76
+ negation, functional expression	0.45	0.66	0.56	0.71	0.64	0.71
	H24	H25	H26	H27	H28	average
Separation of Requirement/Effect	0.58	0.58	0.50	0.49	0.40	0.5145
+ negation	0.69	0.68	0.57	0.57	0.46	0.6090
+ negation, functional expression	0.43	0.68	0.59	0.62	0.47	0.6109

Table 4: Results of Similarity-based Model

6.2 Experimental Results

Similarity-based Model

The results of the similarity-based model are shown in Table 6.2. Taking negative expressions and also taking both negative and functional expressions into consideration improve the accuracy.

Attention Model

Table 6.2 shows the results of the three Attention models. The first column shows the name of models, and the remaining columns show activation function, optimizer, word segmentation criteria, the results on the development set, and the results on the test set at the setting in which the dev set achieves the maximum value.

We tested the same experiments without using word2vec, constructing the vectors only with tf-idf, which revealed almost 10 point decrease in performance for all settings. These experiments show the effectiveness of vector representations by word2vec.

	function	optimizer	suffix/func exp	dev	test
NAIST1	sigmoid	AdaDelta	no attachment	0.6351	0.6154
NAIST2	tanh	Adam	suffix attached	0.6351	0.6538
sigMoMWE	sigmoid	MomentumSGD	suffix/func exp attached	0.6351	0.6667

Table 5: Results of Attention Models

Table 6.2 shows that the setting using tanh for activation function, Adam for optimizer and considering both suffixes and functional expressions achieves the best result.

All the results of both models are summarized in Table 6.2.

7 Discussion

7.1 Similarity-based model

Quite a few queries are answered correctly when negative expressions are taken into consideration. The following is an example:

	Accuracy
Similarity-based models	
Separation of Requirement/Effect	0.5145
+ negation	0.6090
+ negation, functional expression (NAIST3)	0.6109
Attention models	
sigAdaDelta (NAIST1)	0.6154
tanhAdamSufix (NAIST2)	0.6538
sigMoMWE	0.6667

Table 6: Results of all Models

Problem: H21-13-U, label = N

t2:

(Pledges can be created over a Thing that cannot be assigned to others.)

Most similar article:

:

(Article 343: Pledges cannot be created over a Thing that cannot be assigned to others.)

Those sentences are similar within the threshold and the system incorrectly answers Yes if the negative expression is not considered. However, if the negative expression that appears at the end of the sentence is considered, "*(can)(cannot)*", the system could answer this problem correctly by switching the truth-value of the selected article. Our strategy to consider the existence of negative expressions in similar sentences produced positive effect.

Furthermore, using word embeddings that consider attachment of suffixes and functional expressions makes similar articles to t2 more similar than the standard word segmentation criterion. In the following example, the similarity values of those sentences become closer when both suffixes and functional expressions are attached in learning word representations. Table 7.1 shows the similarity values (the lower the better).

Problem: H25-11-O, label = Y

t2:

(If a superficiary fails to pay the rent for two or more consecutive years, the landowner may demand the extinction of the superficies.)

Most similar article

:

(Article 276: If an emphyteuta fails to pay the rent for two or more consecutive years, the landowner may demand the extinction of the emphyteusis.)

- The pair of requirement parts:
  –

    (A superficiary fails to pay the rent for two or more consecutive years)

	without	with
Requirement parts	16.2	13.8
Effect parts	4.2	3.7
Total	20.4	17.5

Table 7: Similarity with/without suffix/func exp attachment

–
(An emphyteuta fails to pay the rent for two or more consecutive years)

- The pair of effect parts:

–
(The landowner may demand the extinction of the superficies)

–
(The landowner may demand the extinction of the emphyteusis)

These results show that better vector representations are learned by taking suffixes and functional expressions attached to the matrix phrases. Furthermore, better similarity measure seems to give better effect of consideration of negative expressions.

7.2 Breakdown Analysis of Attention Model

We checked precision and recall for each of Yes/No cases of the queries in the attention models. The results are shown in Table 7.2. This shows that the Attention models give balanced outputs both for Yes/No cases.

	recall	precision
	Y recall	Y precision
NAIST1 sigAdaDelta	0.5667	0.5000
NAIST2 tanhAdamSufix	0.7000	0.5385
	N recall	N precision
NAIST1 sigAdaDelta	0.6458	0.7045
NAIST2 tanhAdamSufix	0.6250	0.7692

Table 8: Recall and precision for Y/N cases

We tested the use of dropout by setting the ratio at 0.2 and 0.5 in combinations of all previous settings. Those tests show that dropout does not contribute better performance.

8 Conclusion

In this paper, we presented two types of models. One uses clue expressions to separate the requirement and effect parts of both civil law articles and queries, and measure the similarities of those parts to make decision. Considering negative expressions and suffix and functional expressions improved the accuracy.

The other model we presented, attention model, gave better performance than the similarity-based model. It uses the attention mechanism as the relevance factors of the articles. Also in this model, the word representation learning by word2vec that take suffixes and functional expressions into consideration gave better results.

8.1 Future Work

The current systems do not use any information of the exceptional parts of the law articles. Exceptional parts usually start with explicit clue expressions and are not so difficult to identify. However, proper understanding and usage of exceptional expressions are not trivial and need be further investigated.

Acknowledgements

This work was supported by JST CREST Grant Number JPMJCR1513, Japan.

References

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *ICLR*, 2015.

[2] Kiyoun Kim, Seongwan Heo, Sungchul Jung, Kihyun Hong, and Young-Yik Rhim. An ensemble based legal information retrieval and entailment system. *Tenth International Workshop on Juris-informatics (JURISIN)*, 2016.

[3] Mi-Young Kim, Ying Xu, Yao Lu, and Randy Goebel. Legal question answering using paraphrasing and entailment analysis. *Tenth International Workshop on Juris-informatics (JURISIN)*, 2016.

[4] Taku Kudo, Kaoru Yamamoto, and Yuji Matsumoto. Applying conditional random fields to japanese morphological analysis. *Proceedings of Empirical Method for Natural Language Processing 2004*, 230–237, 2004.

[5] M. J. Kusner, Kolkin Sun, Y., N. I., and K. Q Weinberger. From word embeddings to document distances. *ICML*, 15:957–966, 2015.

[6] Suguru Matsuyoshi, Satoshi Sato, and Takehiko Utsuro. A dictionary of japanese functional expressions with hierarchical organization (in japanese). *Journal of natural language processing*, 14(5):123–146, 2007.

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

[8] Ryosuke Taniguchi and Yoshinobu Kano. Legal yes/no question answering system using caserole analysis. *Tenth International Workshop on Juris-informatics (JURISIN)*, 2016.

[9] Tatsuhiko Tsunoda, Hitoshi Shimizu, and Makoto Nagao. A method of extracting conditional - part and effect - part from legal sentences on a compendium of laws using surface clues (in japanese). *IPSJ Natural Language Processing*, 1997(4):129–136, 1997.

[10] Jason Weston, Sumit Chopra, and Antoine Bordes. Memory networks. *CoRR*, abs/1410.3916, 2014.