Proceedings of the Twelfth International Workshop on Juris-informatics (JURISIN 2018)

hosted by the JSAI International Symposia on AI

Workshop Chair

Katsumi Nitta

Raiosha, Keio University, Yokohama, Japan,
November 12-13, 2018

Preface

This volume contains the papers presented at JURISIN2018: Twelfth International Workshop on Juris-informatics (JURISIN 2018) held on November 12-13, 2018 in Tokyo.

Juris-informatics is a new research area which studies legal issues from the perspective of informatics. The purpose of this workshop is to discuss both the fundamental and practical issues among people from the various backgrounds such as law, social science, information and intelligent technology, logic and philosophy, including the conventional "AI and law" area. JURISIN 2018 is the 12th International Workshop on Juris-informatics, which is held in association with the International Symposia on AI by Japanese Society of Artificial Intelligence (JSAI-isAI).

This year, we have twenty-one submissions and each paper was reviewed by three program committee members and as a result, sixteen papers are selected for a presentation at the workshop. Papers cover various field of juris-informatics such as logical inference, natural language processing, and legal issues to use AI in society.

In the second day of JURISIN2018, papers of the Fifth Competition on Legal Information Extraction/Entailment (COLIEE-2018) are presented. The motivation for the competition is to help create a research community of practice for the capture and use of legal information.

Furthermore, as invited speakers, we have Harumichi Yuasa from the Institute of Information Security, Japan and Douglas Walton from the University of Windsor, Canada.

Finally, we would like to thank all the authors who submitted papers and the members of PC for reviewing the submitted papers.

November 12-13, 2018 Katsumi Nitta
Yokohama, Japan

Program Committee

Natasha Alechina	University of Nottingham
Ryuta Arisaka	University of Perugia
Kristijonas Cyras	Imperial College London
Mehdi Dastani	Utrecht University
Marina De Vos	University of Bath
Juergen Dix	Clausthal University of Technology
Phan Minh Dung	AIT
Megumi Fujita	National Institute of Informatics
Randy Goebel	University of Alberta
Guido Governatori	CSIRO
Tatsuhiko Inatani	Kyoto University
Tokuyasu Kakuta	Chuo University
Yoshinobu Kano	Shizuoka University
Tatsuro Kawamoto	Tokyo Institute of Technology
Mi-Young Kim	Department of Computing Science, U. of Alberta, Canada
Nguyen Le Minh	Graduate School of Information Science, Japan Advanced Institute of Science and Technology
Beishui Liao	Zhejiang University
Brian Logan	University of Nottingham
Hatsuru Morita	Tohoku University
Yoichi Motomura	Tokyo Institute of Technology
Makoto Nakamura	Niigata Institute of Technology
Konatsu Nishigai	Tokyo Metropolitan University
Yoshiaki Nishigai	Nihon University
Katsumi Nitta	National Institute of Informatics
Paulo Novais	University of Minho
Julian Padget	University of Bath
Monica Palmirani	CIRSFID
Juliano Rabelo	AMII
Seiichiro Sakurai	Meiji Gakuin University
Katsuhiko Sano	Department of Philosophy, Graduate School of Letters, Hokkaido University
Ken Satoh	National Institute of Informatics and Sokendai
Akira Shimazu	JAIST
Fumio Shimpo	Keio University
Kazuko Takahashi	Kwansei Gakuin University
Satoshi Tojo	JAIST
Katsuhiko Toyama	Nagoya University
Robert van Den Hoven van Genderen	Vrije Universiteit Amsterdam
Leon van der Torre	University of Luxembourg
Bart Verheij	University of Groningen
Yueh-Hsuan Weng	Tohoku University
Masaharu Yoshioka	Hokkaido University
Harumichi Yuasa	Institute of Information Security
Thomas Ågotnes	University of Bergen

Additional Reviewers

Navas-Loro, María
Nguyen Truong, Son
Tran, Vu
Yamada, Hiroaki

Table of Contents

Invited Talk

Introducing information communication technology into civil litigation in Japan

Harumichi Yuasa

Deputy President, Professor, Graduate School, Institute of Information Security

In Japan's civil litigation, a large amount of paper exchanges has been carried out for a long time. The reason is that the Japanese Civil Procedure Law employs a paradigm that courts browse paper documents as a method of examining evidence in principle. In addition to paper documents treated as evidence, many of the administrative procedures related to various civil lawsuits rely on paper documents.

In the past, attempts to introduce information communication technology into civil lawsuits have been tried several times.

In 1998, a video conference system for witness interrogation was introduced (Article 204 of Civil Procedure Law). Also in the criminal procedure, a video conference system was introduced in 2001. However, the teleconference system is not necessarily used frequently because of the high communication fee, environment that teleconference system is not installed in all court rooms, etc. In 2004, an online system of dunning proceedings was introduced. However, damages, salaries, rents and other dunning can not be processed via online. In 2008, Sapporo District Court conducted an experiment of online complaint system, but there were little users.

In 2017, the Cabinet decided "Future Investment Strategy 2017". It stipulated that introduction of information communication technology into civil proceedings should be examined for the purpose of implementing efficient and convenient court procedures, and that the conclusion will be published within fiscal year of 2017. Following this, in October 2017, committee to consider introducing information communication technology into court procedures was set up. The committee published the report in March, 2018.

In summary, they propose to introduce information communication technology into civil proceedings in three stages: e-filing, e-court and e-management.

In this session, we will discuss about the three stages in detail such as: whether to realize the three stages at the same time, the relationship between digitalization of evidence and digital forensics, cyber security, protection of privacy, information system design, timing of replacement of existing information systems and online systems, and other problems.

Invited Talk

Logical and Legal Relevance

Douglas Walton

University of Windsor

This paper discusses the relationship between logical relevance and legal relevance in order to help provide a foundation for helping computational argumentation models of relevance move ahead based on some clear and coherent notion of relevance that is, broadly speaking, logical in nature. This argumentation-based notion of relevance is logical, in that it uses forms of argument (argumentation schemes) to judge relevance or irrelevance of an argument (or of a proposition in an argument, such as a premise). But it is also pragmatic, meaning that it needs to take the use of an argument in a communicative context, such as a trial setting, into account.

Some recent computational argumentation systems in AI and law have used the notion of relevance in abstract argumentation systems to provide a way for cutting down the argumentation in a formal dialogue that might go on endlessly. But relevance has long been a contested concept in many disciplines, including law. Peter Tillers, and a series of long footnotes in one edition of J. H. Wigmore's monumental work on evidence law, Evidence in Trials at Common Law, has summarized the not very well-known history of the subject. On Wigmore's theory, logical relevance is the underlying foundation of the legal concept of relevant evidence. He thought there was a science of proof, which constituted a structure of logical reasoning underlying the trial rules, which should be used to judge relevance in legal cases. But what is that logic? One answer is deductive logic, another is the logic of probability (referring to some form of statistical/Bayesian reasoning). This paper will outline a third alternative, a logical and pragmatic notion of relevance derived from recent work in argumentation theory and informal logic. It is shown that this third alternative approach is promising by applying some argumentation tools to some examples where judging relevance or irrelevance is a problem.

The concept of relevance is fundamental to rules of evidence in law, such as the Federal Rules of Evidence, where relevance is defined in terms of an increased probative weight transferred to the conclusion of an argument from its premises. But it remains unclear whether this notion of probative weight is better analyzed by Bayesian methods of probability calculations or along other lines that use inference to the best explanation, where an explanation can be made more plausible by argument supporting it, or less plausible by arguments attacking it.

This paper will consider some legal examples (and one example form a logic textbook) where relevance is an issue, and use argumentation tools and theories to analyze the arguments in them, making special use of argumentation schemes and

argument diagrams. Wigmore himself was one of the founders of the use of argument diagramming (argument mapping). He used evidence charts to evaluate evidential reasoning in some legal cases. This approach died out for a long time, but has now returned to some prominence in AI. Now many formal argumentation models in AI are based on graph structures that are essentially argument diagrams.

At this present point in the development of computational argumentation systems in AI and law, therefore, it is timely to pay more systematic attention to exploring relevance by examining legal cases to show what the problems are in defining and applying some useful notion of relevance. The work in this paper is part of a larger research project on relevance in argumentation that has the essential prerequisite of starting with making some progress on the goal of illuminating the distinction between logical and legal relevance.

On the Legal Debugging in PROLEG program

Wachara Fungwacharakorn[1] and Ken Satoh[2]

[1] National Institute of Informatics, Sokendai University, Tokyo, Japan
wacharaf@nii.ac.jp
[2] National Institute of Informatics, Sokendai University, Tokyo, Japan
ksatoh@nii.ac.jp

Abstract. PROLEG is a legal reasoning system adapted from Prolog. It gives legal reasoning based on rule base extracted from legislation. However, legislation itself contains unexpected exceptions which can be detected only when an exceptional situation happens. In this paper, we proposed a technique called "legal debugging" to find out which rule conditions should be revised in such exceptional cases. We firstly present the legal debugging in the context of non-recursive logic program. Then, we extend into the context of PROLEG program and illustrate legal debugging in PROLEG with a Supreme Court case.

Keywords: Legal Reasoning, Legal Representation, Algorithmic Debugging.

1 Introduction

PROLEG (PROlog based LEGal reasoning support system) [1] is a logic programming adapted from Prolog. It reflects the legal reasoning procedures called ultimate fact theory practiced in Japanese law schools. In brief, PROLEG is similar to Prolog but using exception instead of negation. To simulate legal reasoning, PROLEG retrieves facts from the case and gives reasoning based on legal rules.

However, legal rules itself have unexpected exceptions. They are hard to be detected unless such particular exceptional case happens. For example, let's consider a case that a plaintiff made a lease contract for his house between him and the defendant. The defendant let his son use a room when he comes back home for a while. Then the plaintiff claimed that the contract was ended by his cancellation of the contract for the reason that the defendant subleases without permission.

According to the related piece of legislation, Japanese Civil Code Article 612, the leaser may cancel the contract if the lessee lets any third party use the property without lessor's permission.

> **Phrase 1:** A lessee may not assign the lessee's rights or sublease a leased thing without obtaining the approval of the lessor.
> **Phrase 2:** If the lessee allows any third party to make use of or take profits from a leased thing in violation of the provisions of the preceding paragraph, the lessor may cancel the contract.

If we interpret the legislation literally, the cancellation is valid. However, when we take into account that the person who temporally uses the room was the defendant's son and he did not harm the confidence between a lessee and a lessor, the regulation might look too strict. The Supreme Court agreed to this and decided following [2]:

> Phrase 2 is not applicable in exceptional situations where the sublease does not harm the confidence between a lessee and a lessor, and therefore the lessor cannot cancel the contract unless they prove the lessee's destructing of confidence.

Hence, when giving a legal reasoning in this case, some rule conditions would be affected. Finding those rule conditions can be regarded as legal debugging. A debugger identifies a culprit, a condition that causes unexpected consequences, and allows the users to resolve it.

Therefore, in this paper, we try to formalize the process of legal debugging in PROLEG program. This paper is structured as follows. Section 2 describes legal reasoning characteristics in the context of logic program with negation as failure and defines a legal debugging process in that context. Section 3 extends the legal debugging process into the context of PROLEG. Section 4 demonstrates a legal debugging process in PROLEG using the above Supreme Court case. Section 5 compares legal debugging and other related works on debugging process. The final section summarizes the idea of legal debugging and its application as the future work.

2 Legal Reasoning and Debugging in Logic Program Context

2.1 Legal Reasoning in Logic Program Context

Before defining the legal debugging process, this section introduces basic assumptions of legal reasoning in the context of logic program with negation as failure because it is familiar to many logic programmers. Generally, the logic program with negation as failure is defined by following definitions.

Definition 1. [Rules] Let h, b_i ($1 \leq i \leq n$), and c_j ($1 \leq j \leq m$) be propositions and R be a *rule* of the form $h \leftarrow b_1,\ldots,b_n$, *not* $c_1,\ldots,$ *not* c_m.

We denote h as $head(R)$, $\{b_1,\ldots,b_n\}$ as $pos\_body(R)$, $\{c_1,\ldots,c_m\}$ as $neg\_body(R)$, and $body(R)$ as $pos\_body(R) \cup neg\_body(R)$. We allow a rule without a body denoted as *fact*.

Definition 2. [Satisfaction] Let I be a set of propositions and R be a rule. *I satisfies R* if the following condition is satisfied:

if $pos\_body(R) \subseteq I$ and $neg\_body(R) \cap I = \phi$ then $h \in I$.

Definition 3. [Model] Let P be a finite set of rules and I be a set of propositions. If I satisfies every rule in P, we call that I is a *model* of P. The *minimum model* of P (denoted as $min(P)$) is a model of P which is the minimum in the sense of set inclusion.

Example 1. An example of a logic program P

$p \leftarrow q$, *not* r.
$q \leftarrow f_1$.
$r \leftarrow f_2$.

From Example 1, $\{q\}$ does not satisfy the rule R: $p \leftarrow q$, *not* r because $pos\_body(R) \subseteq \{q\}$ and $neg\_body(R) \cap \{q\} = \phi$ but $p \notin \{q\}$. The example sets of propositions that satisfy every rule in P are ϕ, $\{r\}$, $\{q, r\}$, $\{p, q\}$ hence they are models of P and the minimum model of P is ϕ ($min(P) = \phi$). Sometimes the minimum model is referred as an answer set.

Note that propositions in this paper do not contain arguments (Zeroth-order logic). Propositions that contain arguments are such as $p(a)$, or $p(a,b)$ which can be seen in legal representation such as *contract*(*plaintiff*, *defendant*) but we omit them in this paper to simplify the idea. We will continue to work on them in the future works.

There are also two assumptions from legal reasoning that we use for simplifying and distinguishing the logic program that represents legal knowledge from general logic programs. First assumption from legal reasoning is that, generally, the rules from legislation are not recursive. Recursive rules are those rules whose dependency graph contains a loop (see details in [3]). For example $\{p \leftarrow p.\}$,$\{p \leftarrow$ *not* $p.\}$, $\{p \leftarrow q.\ q \leftarrow p.\}$ are recursive, but the program in Example 1 is not recursive .

Second assumption is that, some propositions in legislation are *contingent*. Their truth values shall not be determined analytically but by directly derive from the evidences e.g. the signature is valid or not. Hence, such propositions never appear as a head of a rule with a body so their truth values could be changed when modifying other propositions so we could define *contingent* as follows.

Definition 4. [Contingency] A proposition is *contingent* if it does not appear as a head of a rule with a body.

From Example 1, f_1 and f_2 are contingent. Their truth value could not be changed by modifying other propositions but only by adding them as *facts* (a rule without a body). In contrast, p, q, or r could be changed their truth value by modifying other propositions, for example, p could become true if we made q true (such as just adding q as a *fact*) and kept r false.

However, some propositions could not be true concurrently. From Example 1, if r was true, p could not be true. We could say that $\{r\}$ (or any sets that contain r) does not support p. To determine this relation between rules and a set of propositions, the idea of *support* is defined as follows.

Definition 5. [Support] A set of propositions *S supports* a non-contingent proposition *p* w.r.t. a logic program *P* if there is a rule $R \in P$ such that *p* is the head of the rule (p=*head*(R)), *pos_body*(R) $\subseteq S$, and *neg_body*(R) $\cap S = \phi$. *R* is called a supporting rule of *p* w.r.t. *S*.

Theorem 1. [Relation between model and support] Given *P* as a non-recursive logic program and *p* is non-contingent and, $p \in$ *min*(*P*) if and only if *min*(*P*) supports *p*.

Proof.
Suppose *min*(*P*) supports *p* but $p \notin$ *min*(*P*), then there is a rule $R \in P$ supporting of *p*.
Hence p=*head*(R), *pos_body*(R) $\subseteq$ *min*(*P*) and *neg_body*(R) $\cap$ *min*(*P*) = ϕ.
But since $p \notin$ *min*(*P*), *min*(*P*) does not satisfy *R*.
It leads to contradiction with the definition of model.
Suppose *min*(*P*) does not support *p* but $p \in$ *min*(*P*), then no rules are supporting *p*.
Thus, *min*(*P*) - {*p*} must be a model because it satisfies every rule in *P*.
It leads to contradiction with the definition of the minimum model.

2.2 Legal Debugging in Logic Program Context

When the direct interpret of legislation gives unexpected results, it means that we intend a different minimum model from the existed minimum model. However, because truth values of contingent propositions are already finalized, one that is contingently true would not be possible to be excluded from a minimum model and that is contingently false would not be possible to be included in a minimum model.

From Example 1, we could not intend $\{f_1\}$ to be a minimum model because f_1 is contingently false and we could not change its truth value. In contrast, we could intend $\{q\}$ to be a minimum model because *q* is not contingent so we allow changing the truth value of *q* (such as just adding *q* as a *fact*). Therefore, we define an intended model denoted as *IM* with following definitions.

Definition 6. [Intended model constraint] A set of propositions *IM* can be an intended model of a logic program *P* if and only if a set of contingent facts in *min*(*P*) and a set of contingent facts in *IM* are equal.

Consequently, if *IM* is different from *min*(*P*), there must be a non-contingent *p* that is not currently derived but intended to be derived ($p \notin$ *min*(*P*) but $p \in$ *IM*) or currently derived but intended not to be derived ($p \in$ *min*(*P*) but $p \notin$ *IM*). Such proposition is called *unexpected* which is defined as follows.

Definition 7. [Unexpectation] A proposition *p* is unexpected w.r.t. an intended model *IM* and a logic program *P* if $p \notin$ *min*(*P*) but $p \in$ *IM* or if $p \in$ *min*(*P*) but $p \notin$ *IM* . Contingent propositions could not be unexpected due to constraints in Definition 6.

To modify the program so its minimum model was changed to what we intended, we must work on some non-contingent propositions called *culprit* defined as follows.

Definition 8. [Culprit] a non-contingent proposition p will be a *culprit* of a logic program P w.r.t. an intended model IM if

- $p \in IM$ but IM does not support p or
- $p \notin IM$ but IM supports p

From Example 1, if $IM = \{q\}$.

- $p \notin IM$ but IM supports p by $p \leftarrow q$, *not* r, hence p is a culprit of P
- $q \in IM$ but IM does not support q, hence q is a culprit of P
- $r \notin IM$ and IM does not support r, hence r is not a culprit of P

Then, we get the following property.

Theorem 2. [Unexpectation Propagation] Given P as a non-recursive logic program and IM as an intended model that not equal to $min(P)$. If p is unexpected w.r.t. IM and P, then

- there is existed another proposition q in a body of a rule whose head is p such that q is unexpected w.r.t. IM and P otherwise
- p is a culprit w.r.t. IM and P itself

Proof.
If $p \in IM$ and p is also not a culprit
Then there is a rule R supporting of p w.r.t. IM.
Hence, $pos\_body(R) \subseteq IM$ and $neg\_body(R) \cap IM = \phi$.
But $p \notin min(P)$, R is not a supporting rule of p w.r.t. $min(P)$.
Thus, $pos\_body(R) \not\subset min(P)$ or $neg\_body(R) \cap min(P) \neq \phi$.
Because P is non-recursive, $p \notin body(R)$
Hence, there is another proposition $q \in pos\_body(R)$ such that $q \in IM$ and $q \notin min(P)$ or another proposition $r \in neg\_body(R)$ such that $r \notin IM$ and $r \in min(P)$ that could be found in a body of a rule R whose head is p .

If $p \notin IM$ and $p \in min(P)$, there is a rule R supporting of p w.r.t. $min(P)$.
Hence, $pos\_body(R) \subseteq min(P)$ and $neg\_body(R) \cap min(P) = \phi$.
And if p is not a culprit, R is not a supporting rule of p w.r.t. IM.
Thus, $pos\_body(R) \not\subset IM$ or $neg\_body(R) \cap IM \neq \phi$.
Because P is non-recursive, $p \notin body(R)$
Hence, there is another proposition $q \in pos\_body(R)$ such that $q \notin IM$ and $q \in min(P)$ or another proposition $r \in neg\_body(R)$ such that $r \in IM$ and $r \notin min(P)$ that could be found in a body of a rule R whose head is p.

Example 2. Given P a logic program as in Table 1, then f is contingent and $min(P) = \{p, r, f\}$. Let the intended model IM be $\{q, f\}$, p, q, and r are unexpected, and r is a culprit according to Theorem 2.

Table 1. An illustrated example of Theorem 2

Rules	Unexpected found in a head	Supporting Rule w.r.t. $min(P)$	Supporting Rule w.r.t. IM	Unexpected found in a body
$p \leftarrow not\ q.$	p	Supporting	-	q
$p \leftarrow not\ f.$	p	-	-	-
$q \leftarrow not\ r.$	q	-	Supporting	r
$r \leftarrow f.$	r	Supporting	Supporting	- (thus r is a culprit w.r.t. IM)
$f.$	-	-	-	-

From Theorem 2, if we query from any unexpected proposition and find another unexpected proposition recursively, as long as the query sequence is finite and does not loop (e.g. P is a finite non-recursive logic program), the query finally succeeds by finding a culprit. Therefore, we could design the finding culprit algorithm as Table 2.

Table 2. Finding culprit algorithm

```
Input: a finite non-recursive logic program P,
an intended model IM, an unexpected proposition p
```

```
Find_culprit(p)
begin
  Find R as a supporting rule of p w.r.t. IM
  if[1] p ∈ IM and there is no such R return p;
  else if p ∈ IM
    Find q ∈ pos_body(R) s.t. q ∈ IM [2] and q ∉ min(P)
    if there is such q return Find_culprit(q)
    Find r ∈ neg_body(R) s.t. r ∉ IM [2] and r ∈ min(P)
    if there is such r return Find_culprit(r)
    return[3] p

  if[1] p ∉ IM and there is such R  return p;
  else if p ∉ IM
    Find R' as a supporting rule of p w.r.t. min(P)
    Find q ∈ pos_body(R') s.t. q ∉ IM and q ∈ min(P)[2]
    if there is such q return Find_culprit(q)
    Find r ∈ neg_body(R') s.t. r ∈ IM and r ∉ min(P)[2]
    if there is such r return Find_culprit(r)
    return[3] p
end
```

[1] They follow directly by the definition of a culprit, Definition 8.

[2] These conditions are actually determined when finding a supporting rule. They are mentioned to emphasize that they are unexpected.

[3] If we could not find another unexpected proposition, then p itself is a culprit according to Theorem 2.

The algorithm is only used for finding one culprit. However, there are cases that a logic program has two culprits or more, or the resolution of a culprit will create another culprit. In such cases, a user has to repeat the same procedure for the revised rules. Actually, it is safe to check that the result minimum model from the revised program is equal to the intended model because it is possible to have the chain reflect from the resolution.

For example, from example 1, if an intended model was $\{q\}$, we can see that only q would be unexpected hence it would be a culprit. If we resolved by just adding q as a fact, $\{p, q\}$ would become a minimum model instead of $\{q\}$ that we expected. In another round, p would become unexpected w.r.t. $\{q\}$ and hence p would be a culprit. If we resolved by removing a rule $p \leftarrow q$, *not* r from the program, $\{q\}$ would finally become the minimum model as we expected.

3 Legal Debugging in PROLEG

PROLEG is different from Prolog in manipulation of negative conditions but the representation power of PROLEG is same as Prolog [4]. In this section, we provide definitions for PROLEG and extend the legal debugging process into the context of PROLEG program. First, these are basic definitions of PROLEG program.

Definition 9. [PROLEG] A PROLEG program P is a pair $\langle H, E\rangle$ where

- H is a set of rules R of the from $h \leftarrow b_1,\ldots,b_n$. where h and b_i $(1 \leq i \leq n)$ are propositions (note that there are no negations in the rule). We denote h as $head(R)$ and $\{b_1,\ldots,b_n\}$ as $body(R)$.
- E is a set of exceptions of the form $exception(h, e)$ where h and e be propositions (note that e is a proposition, not a set of propositions).

Definition 10. [Applicable rule] Let M be a set of propositions. We denote *a set of applicable rules* w.r.t. M, $H^M = \{R \in H \mid \neg\exists exception(head(R), e) \in E \text{ s.t. } e \in M\}$

So if an exception of rule exists ($e \in M$) then the rule is inapplicable.

Definition 11. [Extension] A set of propositions M is an extension of P if M is the minimum model of H^M ($M = \min(H^M)$).

Example 3. *P'* is a PROLEG program.

$p \leftarrow q.$
$q \leftarrow f_1.$
$r \leftarrow f_2.$
exception(p,r).

Let $M = \{r\}$, $H^M = \{ q \leftarrow f_1, r \leftarrow f_2 \}$ ($p \leftarrow q$ is inapplicable here because there is an *exception(p,r)* and $r \in M$). Since $\min(H^M) = \phi$. M is not an extension of P'.

Let $M = \phi$, $H^M = \{ p \leftarrow q, q \leftarrow f_1, r \leftarrow f_2 \}$. Since $\min(H^M) = \phi$. M is an extension of P'.

Actually, Example 3 is the PROLEG representation of the Example 1. We can see that PROLEG representation is actually aligned with the logic program with negation as failure but using exception instead of negation. However, one particular different point is that if we add an exception to a condition, it applies to all rules on that condition unlike a logic program with negation as failure whose negations must be added to the rule one by one as illustrated in Table 3.

Table 3. An equivalent representation between PROLEG (left) and Prolog (right)

$p \leftarrow q$. $p \leftarrow r$. *exception*(p, e).	$p \leftarrow q$, *not* e. $p \leftarrow r$, *not* e.

Because the representation power of PROLEG is same as the logic program with negation as failure, we can use the same idea of supports, culprits, and finding culprit theorem by using an extension of P instead of $min(P)$. For example, these are definition of support (Definition 5), definition of intended model (Definition 6), and definition of unexpectation (Definition 7) in PROLEG.

Definition 12. [Support in PROLEG] A set of proposition S *supports* a proposition p w.r.t. a PROLEG program P if there is a rule R such that $p=head(R)$, $body(R) \subseteq S$, and there is no *exception(p,e)* $\in E$ such that $e \in S$ We call that R is a supporting rule of p w.r.t. S.

Definition 13. [Intended Model in PROLEG] A set of propositions *IM* can be an intended model of a PROLEG program P if and only if a set of contingent facts in an extension of P and a set of contingent facts in *IM* are equal.

Definition 14. [Unexpected in PROLEG] p is unexpected w.r.t. an intended model *IM* and a PROLEG program P if p is not in an extension of P but in *IM* or if p is in an extension of P but not in *IM*.

We design a finding culprit algorithm in PROLEG as shown in Table 4. It is still based on recursion from an unexpected proposition according to Theorem 2. Because the input PROLEG program P is not recursive, it can be deduced that there is only one extension of P.

Table 4. Finding culprit algorithm in PROLEG

```
Input: a finite non-recursive PROLEG logic program P = ⟨H,E⟩,
an intended model IM, an unexpected proposition p
```

```
Find_culprit(p)
begin
  Find R as a supporting rule of p w.r.t. IM
  if p ∈ IM and there is no such R return p;

  else if p ∈ IM
    Find q ∈ body(R) s.t. q is not in an extension of P
    if there is such q return Find_culprit(q)
    Find e s.t. exception(p,e) ∈ E and e is in an extension of P
    if there is such e return Find_culprit(e)
    return p

  if p ∉ IM and there is such R   return p;
  else if p ∉ IM
    Find R' as a supporting rule of p w.r.t. an extension of P
    Find q ∈ body(R') s.t. q ∉ IM
    if there is such q return Find_culprit(q)
    Find e s.t. exception(p,e) ∈ E and e ∈ IM
    if there is such e return Find_culprit(e)
    return p
end
```

4 Example of Legal Debugging in PROLEG

This section illustrates how to make legal reasoning and legal debugging in PROLEG program from the case of Japanese Civil Code Article 612 as mentioned in the introduction section. First, let us introduce the PROLEG representation of Japanese Civil Code Article 612 in Table 5.

Table 5. PROLEG representation of Japanese Civil Code Article 612

```
cancellation_due_to_sublease <=
  effective_lease_contract,
  effective_sublease_contract,
  using_leased_thing,
  manifestation_cancellation.

effective_lease_contract <=
  agreement_of_lease_contract,
  handover_based_on_the_lease_contract.

effective_sublease_contract <=
  agreement_of_sublease_contract,
  handover_based_on_the_sublease_contract.

exception(cancellation_due_to_sublease,approval_of_sublease).
```

```
approval_of_sublease <=
  approval_of_sublease_before_the_day.
```

```
// Facts that are contingently true
   agreement_of_lease_contract.
   handover_based_on_the_lease_contract.
   agreement_of_sublease_contract.
   handover_based_on_the_sublease_contract.
   using_leased_thing.
   manifestation_cancellation.
   nonabuse_of_confidence.
```

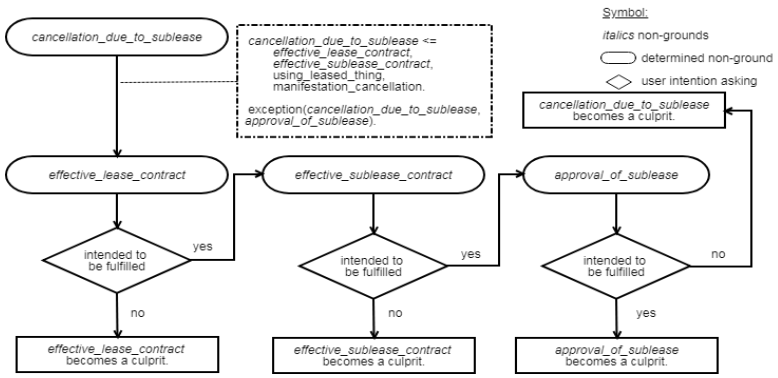


Fig. 1. Legal debugging steps from the rule base

From the rule base, `cancellation_due_to_sublease` is fulfilled because all positive conditions are met and no exception is executed. However, since we concern that the result is too strict, we could initiate debugging by using `cancellation_due_to_sublease` as an unexpected proposition as shown in Fig. 1.

The debugger firstly traced into the supporting rule of `cancellation_due_to_sublease` (the first rule) to determine two conditions in the body `effective_lease_contract` and `effective_sublease_contract`. The debugger asked user whether both conditions were intended to be fulfilled or not. If one of them was intended to be not fulfilled, it became a culprit because the intended model would support it. If both of them were intended to be fulfilled, the debugger retraced on `approval_of_sublease` which is an exception of `cancellation_due_to_sublease`. Then, the debugger asked user that `approval_of_sublease` was intended to be fulfilled or not. If it was intended to be fulfilled, it became a culprit because the intended model would not support it. If `approval_of_sublease` was intended to be not fulfilled, then there was no unexpected condition for `cancellation_due_to_sublease`, hence `cancellation_due_to_sublease` became a culprit itself.

After a culprit was determined, the resolution of culprit could be introduced. For example, to determine this case according to the Supreme Court decision, it is still intended that `effective_lease_contract` and `effective_sublease_contract` to be fulfilled and `approval_of_sublease` to be not fulfilled. Hence the `cancellation_`

`due_to_sublease` becomes a culprit. Then, the debugger could ask the user to resolve the culprit. Generally, it could be resolved by introducing some unconcerned facts as an exception of the culprit. In this case, `exception (cancellation_due_to_sublease, nonabuse_of_confidence)` shall be introduced to solve the unexpected result. This resolution also corresponds to the Supreme Court decision.

5 Discussion and Related works

The idea of legal debugging is closely related to the idea of debugging logic program which has been widely concerned since the logic program was established [5] such as the notable work of Ehud Shapiro's algorithmic program debugging in 1982 [6]. The idea has been developed in many logic programming perspectives, especially for those in answer set programming. Perhaps the idea of debugging answer set programming was introduced in [7] before many types of unexpected results has been concerned such as inconsistency [8, 9] and absence of the answer sets [10, 11]. Many different techniques have also been proposed to solve different applications such as stepping framework [12] and meta-programming [13].

In this paper, the legal representation is aligned with answer set program with negation as failure but different in some perspectives. First, the program is assumed to be not recursive so we can eliminate the problem from loops such in [8–11]. Second, some propositions are contingent of evidences. Therefore, even the concept of *support* is aligned with [7] but the logic program was restricted to not be able to touch truth value of contingent facts. Finally, the user does not recognize all intended propositions at the first time but rather recognize them when responding the debugger steps by steps. Hence the algorithm was designed to be in steps with those three specific assumptions.

6 Conclusion and Future Works

In this paper, we have proposed the idea of *legal debugging* especially in PROLEG program. The idea has been presented in non-recursive program which we assume that some propositions' truth values shall not be changed (called *contingent* facts). Then, we have proposed the idea of *culprit*, a rule condition that needs to be revised when direct legal reasoning from legislation gives unexpected result. We begin the debugging process from the unexpected result as an initial *unexpected* proposition. The debugger follows a sequence of unexpected propositions until it meets a culprit otherwise the initial unexpected proposition is a culprit itself. We prove the correctness of algorithm under non-recursive logic programing with negation as failure, and then we extend the algorithm to PROLEG program. In future, we will extend the algorithm for first-order logic programs with arguments and develop an interactive debugger in PROLEG program which asks user intention and steps into rule base to find culprits similarly to the debugging process in programming.

References

1. Satoh, K., Asai, K., Kogawa, T., Kubota, M., Nakamura, M., Nishigai, Y., Shirakawa, K., Takano, C.: PROLEG: An Implementation of the Presupposed Ultimate Fact Theory of Japanese Civil Code by PROLOG Technology. In: New Frontiers in Artificial Intelligence. pp. 153–164. Springer, Berlin, Heidelberg (2010).
2. Minshu Vol. 50, No. 9 at 2431: 1994 (O) 693. (1996).
3. Ullman, J.: Principles of database and knowledge-base systems. Computer Science Press (1988).
4. Satoh, K., Kogawa, T., Okada, N., Omori, K., Omura, S., Tsuchiya, K.: On Generality of PROLEG Knowledge Representation. In: Proceedings of the 6th International Workshop on Juris-informatics (JURISIN 2012), Miyazaki, Japan. pp. 115–128 (2012).
5. Caballero, R., Riesco, A., Silva, J.: A Survey of Algorithmic Debugging. ACM Comput. Surv. 50, 60:1–60:35 (2017).
6. Shapiro, E.Y.: Algorithmic Program Debugging. MIT Press, Cambridge, MA, USA (1983).
7. Brain, M., De Vos, M.: Debugging Logic Programs under the Answer Set Semantics. In: Answer Set Programming (2005).
8. Syrjänen, T.: Debugging inconsistent answer set programs. In: Proc. NMR. pp. 77–83 (2006).
9. Schulz, C., Satoh, K., Toni, F.: Characterising and Explaining Inconsistency in Logic Programs. In: Calimeri, F., Ianni, G., and Truszczynski, M. (eds.) Logic Programming and Nonmonotonic Reasoning. pp. 467–479. Springer International Publishing (2015).
10. Caminada, M., Sakama, C.: On the existence of answer sets in normal extended logic programs. ECAI 2006 Proc. 141, 743–744 (2006).
11. Oetsch, J., Pührer, J., Tompits, H.: Catching the ouroboros: On debugging non-ground answer-set programs. Theory Pract. Log. Program. 10, 513–529 (2010).
12. Oetsch, J., Pührer, J., Tompits, H.: Stepping through an answer-set program. In: International Conference on Logic Programming and Nonmonotonic Reasoning. pp. 134–147. Springer (2011).
13. Gebser, M., Pührer, J., Schaub, T., Tompits, H.: A meta-programming technique for debugging answer-set programs. In: AAAI. pp. 448–453 (2008).

Reasoning by a Bipolar Argumentation Framework for PROLEG

Tatsuki Kawasaki, Sosuke Moriguchi, and Kazuko Takahashi

School of Science&Technology, Kwansei Gakuin University,
2-1, Gakuen, Sanda, 669-1337, JAPAN
dxk96093@kwansei.ac.jp, chiguri.s@acm.org, ktaka@kwansei.ac.jp

Abstract. We develop a system allowing lawyers and law school students to analyze court judgments. We describe a transformation from the logic programming language PROLEG to a bipolar argumentation framework (BAF) and the legal reasoning involved. Legal knowledge written in a PROLEG program is transformed into a BAF, in which the structure of argument in a judgment is clear. We describe two types of reasoning by the BAF: the entire structure and causality of arguments, and identification of the evidence required.

Keywords: bipolar argumentation framework, PROLEG, reasoning, semantics

1 Introduction

Recently, information technology and artificial intelligence are now vigorously applied in various fields, including those that have not yet been fully digitized or automated. In the context of legal reasoning, although the use of artificial intelligence has attracted a great deal of attention, higher-level and more practical support exploiting recent technological developments is required. Deriving support for a judgment is important. When seeking to support a judgment, it is essential to develop a system that can be easily used by lawyers who are not computer scientists; also, the system must be highly reliable and must reason accurately and rapidly. Lawyers must be able to access the system in a straightforward manner, and the system must both describe the process leading to judgment and the way in which the law was applied.

In terms of the former consideration, as law is logical, it is reasonable to base the system on such logic and reason from that perspective. Several legal reasoning systems have adopted logic programming such as Prolog as their descriptive languages. However, it is difficult for a lawyer who is not familiar with computer science to directly write Prolog code. A PROLEG system was developed to solve this problem [17]. The system was designed to support inferences based on the Japanese Presupposed Ultimate Facts Theory (termed "Yoken-jijitsu-ron" in Japanese) of the Japanese civil code, and it is currently applied to the Japanese penal code. The theory deals with uncertainties that sometimes arise in court, where a judge must give a decision even if evidence is lacking. The PROLEG

language is an extension of Prolog. Each presupposed ultimate fact is represented using general rules written in the form of *if-then* statements and exceptions. Exceptions of fact apply to all general rules and are used as court defenses. The use of exceptions rather than negative atoms creates a structure equivalent to that of a law, allowing lawyers to intuitively understand the program. An interpreter is implemented in Prolog. A burden-of-proof [14] is attached to each ultimate fact to allow for decision-making even if the fact is not proven to be true. This is achieved using the negation-as-failure inference of Prolog; thus, for a given goal, a general rule is applied and the goal is true unless there is an exception.

In terms of legal process and application, it is appropriate to employ argumentation to describe both the judgment process and how the law was applied [1]. An argumentation system reveals both the causality of arguments (for example, how arguments interacted to create a judgment) and the influence of evidence. Argumentation is powerful when used to resolve conflicts, not only formalizing the structure of the process but also incorporating any uncertainties.

In the time since Dung proposed the abstract Argumentation Framework (AF) [9], many extensions and revisions of the system have been published [15]. AF represents an argumentation by a pair of a set of arguments and a set of attacks between arguments, ignoring the contents of arguments. Several AF semantics have been defined; acceptable arguments are calculated based on these semantics. Visualization tools appropriate for argumentation systems have also been developed (e.g., [16]).

Although PROLEG facilitates the representation of a law, it is difficult to grasp the judgment process or argument causality from the execution trace. On the other hand, although it is possible to create an AF representing the interaction between a plaintiff and a defendant in court, it is difficult to directly write the structure of a law per se, or the part of the law used to create an argument in an AF form. Therefore, we combined the two systems.

We developed a transformation from PROLEG to a bipolar argumentation framework (BAF) [6], an extended AF, and showed its correctness [11]. More specifically, we gave a semantics for the BAF obtained as a result of the transformation, and proved that the answer set of the PROLEG program was the same as the set of acceptable BAF arguments. Here, we describe how the BAF reasons.

Consider the following PROLEG program representing the penal code that defines the "crime of murder."[1] The first clause indicates the general rule and the second clause an exception. The text states that if the object is a human (not a dead body) and there exists both the action of murder and also the intention to murder, then the crime of murder has been committed unless there is a legitimate defense.

```
crime_of_murder <= human, action_of_murder, intention_to_murder.
exception(crime_of_murder, legitimate_defense).
```

[1] Note that the examples shown here are simplified versions of the actual penal code; the conditions per se are simplified and the legal terminology is not precise.

When evidence is provided, the facts on which that evidence bears are proved, and it is then decided whether the crime_of_murder has been committed or not.

A judge should explain the judgment process to persuade those concerned with the transparency of justice. In such a legal situation, what is required is not only the outcome of judicial reasoning but also an explanation of the reasoning process or the cause-and-effect relationships of arguments used in reasoning. For example, if the crime_of_murder was adjudged to not in fact have been committed; this may be because of a lack of evidence of intention_to_murder, or because a legitimate_defense was available.

Our transformed BAF not only shows the process and structure of judgment, but also suggests a strategy by which a user can achieve a desired goal. If a defendant/plaintiff wishes to argue that a law should or should not be applied, the BAF identifies the evidence that must be presented and any counter-arguments that may arise. For example, when a prosecutor wishes to charge the crime_of_murder, but finds that the lack of intention_to_murder is a complicating factor, s/he will look harder for evidence of intention_to_murder. Here, we discuss such reasoning on our BAF.

This paper is organized as follows. In Section 2, we briefly explain PROLEG. In Section 3, we describe the BAF that we develop and its semantics. In Section 4, we develop the transformation rule from PROLEG to BAF. In Section 5 we describe how the BAF reasons. In Section 6, we compare our method with those of others. Finally, in Section 7, we offer conclusions and describe our planned future work.

2 Legal Description Language: PROLEG

The PROLEG program P is defined as pairs of $\langle \mathcal{R}, \mathcal{E} \rangle$, a finite set of rules, and a finite set of exceptions. Each rule is a Horn clause of the form $H \Leftarrow B_1, \ldots B_n$, where $H, B_1, \ldots, B_n$ are atoms (n may be 0; we term such a rule *a fact rule* or simply *a fact*). Each exception is in the form exception(H, B).

A fact is something given as an evidence in a court case, whereas the other rules and exceptions describe the general case. That is, the former (the facts) are generally given in an instantiated form whereas the other rules and exceptions include variables. In the examples that follow, we use a proposition for simplicity.

For each rule R or exception E, we employ the functions *head* and *body* such that $head(R) = H$ and $body(R) = \{B_1, \ldots, B_n\}$ if $R = H \Leftarrow B_1, \ldots, B_n$; $head(E) = H$ and $body(E) = \{B\}$ if $E =$ exception(H, B). An atom may be defined by more than one rule. This means that there may exist R_1 and R_2 $(R_1 \neq R_2)$ such that $head(R_1) = head(R_2)$.

Example 1. The following is an example of a PROLEG program.

```
p <= q1, q2.
exception(q1, r).
q2 <=.
r <=.
```

The semantics of the PROLEG program P are defined as an answer set (a set of ground atoms). M is the *answer set* of P iff M is the minimum model of the set of Horn clauses, $\{R \in \mathcal{R} \mid \forall E \in \mathcal{E},$ if $head(E) = head(R)$ then $body(E) \not\subseteq M\}$. The expressive power of PROLEG is the same as that of a normal logic program with an answer set [10, 18].

PROLEG allows cyclic definitions. However, here, we deal with an acyclic PROLEG program, because the Japanese civil and penal codes are usually written in an acyclic manner.

3 Bipolar Argumentation Framework

First, we define our argumentation framework.

Definition 1 (argumentation framework). An argumentation framework *is defined as a pair* $\langle AR, AT\rangle$ *where* AR *is the set of arguments and* AT *is a binary relation on* AR*, termed* an attack*. If* $(A, A') \in AT$*, we state that* A attacks A'*.*

A BAF is an extension of an AF in which the two relations of attack and support are defined over a set of arguments [6]. We define a support relation between a power set of arguments and a set of arguments; this differs from the usual BAF format, because the body of a rule generally includes more than one atom in PROLEG.

Definition 2 (bipolar argumentation framework). *A BAF is defined as a triple* $\langle AR, ATT, SUP\rangle$ *where* AR *is a finite set of arguments,* $ATT \subseteq AR \times AR$ *and* $SUP \subseteq (2^{AR} \setminus \{\emptyset\}) \times AR$*. We denote* $\mathrm{att}(B, A)$ *if* $(B, A) \in ATT$*, and* $\mathrm{sup}(\mathbf{A}, A)$ *if* $(\mathbf{A}, A) \in SUP$*.*

Example 2. Figure 1 is a graphical representation of a bipolar argumentation framework $\langle\{a, b, c, d, e\},\ \{(b, a), (e, d)\},\ \{(\{c, d\}, a)\}\rangle$. In the figure, the straight arrow indicates an attack relation and the wavy arrow a support relation.

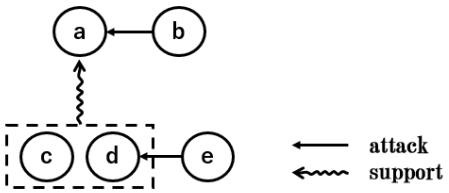


Fig. 1. Example of BAF.

We derived semantics for the BAF based on labeling [5]. Usually, labeling is a function from a set of arguments to $\{in, out, undec\}$, but *undec* is unnecessary here, because the BAF is acyclic. An argument labeled *in* is considered to be an accepted argument.

Definition 3 (labeling). *For* $\langle AR, ATT, SUP\rangle$, labeling $\mathcal{L}$ *is a function from* AR *to* $\{in, out\}$.

Labeling of a set of arguments is denoted as follows: $\mathcal{L}(\mathbf{A}) = in$ if $\mathcal{L}(A) = in$ for all $A \in \mathbf{A}$; $\mathcal{L}(\mathbf{A}) = out$, otherwise.

We use the label *in* to identify arguments that are neither attacked nor supported by any other argument. When an argument is both attacked and supported, the attack is supposed to be stronger than the support. We assign a label *out* to an argument that is attacked by another argument with the label *out*, and simultaneously supported by the set of arguments with the label *out*. Note that an argument lacking support is labeled *out*, even if it is attacked by an argument labeled *out*.

Definition 4 (complete labeling). *For* $\langle AR, ATT, SUP\rangle$, *labeling* $\mathcal{L}$ *is* complete *iff the following conditions are satisfied for any argument* $A \in AR$.

- $\mathcal{L}(A) = in$ *if*
  - $(\forall B \in AR, \neg\mathrm{att}(B, A)) \wedge (\forall \mathbf{A} \subseteq AR, \neg\mathrm{sup}(\mathbf{A}, A))$
    or
  - $(\forall B \in AR, \mathrm{att}(B, A) \Rightarrow \mathcal{L}(B) = out) \wedge (\exists \mathbf{A} \subseteq AR, \mathrm{sup}(\mathbf{A}, A) \wedge \mathcal{L}(\mathbf{A}) = in)$.
- $\mathcal{L}(A) = out$, *otherwise.*

Figure 2 shows the complete labeling of four BAFs.

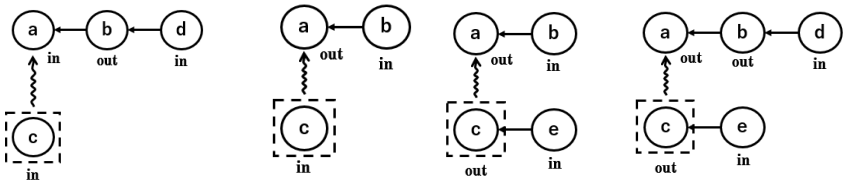


Fig. 2. Examples of BAFs with complete labelings.

Example 3. For a BAF in Figure 1, $\mathcal{L}(b) = \mathcal{L}(c) = \mathcal{L}(e) = in$ and $\mathcal{L}(a) = \mathcal{L}(d) = \mathcal{L}(\{c, d\}) = out$.

The following theorem holds [11].

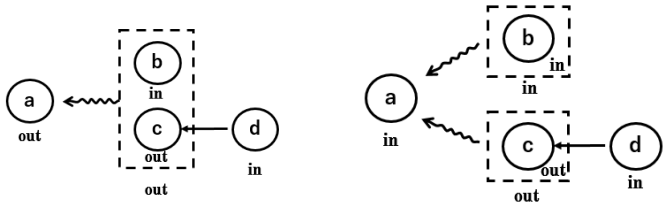
Theorem 1. *For any acyclic BAF, there is exactly one complete labeling.*

Note that we distinguish the case in which an argument is supported by a set of arguments from that in which it is supported by multiple arguments separately.

Example 4. Consider two BAFs baf_1 and baf_2 shown in Figure 3. Formally, baf_1 is described as $\langle\{a, b, c, d\}, \{(d, c)\}, \{(\{b, c\}, a)\}\rangle$ and baf_2 is described as $\langle\{a, b, c, d\}, \{(d, c)\}, \{(\{b\}, a), (\{c\}, a)\}\rangle$.

In baf_1, the argument a has one support that is a set of two arguments, whereas in baf_2, it has two supports, both of which are singletons.

Let $\mathcal{L}_1$ and $\mathcal{L}_2$ be the complete labeling of baf_1 and baf_2, respectively. In baf_1, $\mathcal{L}_1(b) = \mathcal{L}_1(d) = in$ and $\mathcal{L}_1(c) = out$ hold. It follows that $\mathcal{L}_1(\{b, c\}) = out$ holds. Therefore, $\mathcal{L}_1(a) = out$. On the other hand, in baf_2, $\mathcal{L}_2(b) = \mathcal{L}_2(d) = in$ and $\mathcal{L}_2(c) = out$ hold by the similar reasoning. However, $\mathcal{L}_2(\{b\}) = in$ and $\mathcal{L}_2(\{c\}) = out$. Therefore, $\mathcal{L}_1(a) = in$.



(a) baf_1: supported by a set (b) baf_2: supported independently

Fig. 3. Two types of support.

.

4 Transformation

4.1 Transformation rule

We here show a transformation from a PROLEG program to a BAF. The atoms, rules, and exceptions of the PROLEG program are transformed into arguments, supports, and attacks, respectively.

We add two types of arguments to the BAF that do not feature as explicit atoms in PROLEG. One is an argument reflecting the *absence* of any rules of inference in PROLEG. In PROLEG, an atom H that does not appear in the header of any rule or exception is not in the answer set. On the other hand, arguments that are neither attacked nor supported are labeled *in*. To fill this gap, we add the argument $\mathrm{ab}(H)$ that attacks H. We term this argument *an absence argument*. We also add arguments showing the *existence* of fact rules. For a fact rule (i.e., a rule in the form $H \Leftarrow$), there are no arguments that support H in BAF; any support is a binary relation. Therefore, we add an argument $\mathrm{ex}(H)$ that supports H. We term this argument an *existence argument*.

Definition 5 (transformation rule).
Transformation from a PROLEG program $\langle \mathcal{R}, \mathcal{E} \rangle$ *to a BAF* $\langle AR, ATT, SUP \rangle$ *is defined as follows.*

- $Atom = \bigcup_{R \in \mathcal{R}} (\{head(R)\} \cup body(R)) \cup \bigcup_{E \in \mathcal{E}} (\{head(E)\} \cup body(E))$
- $Rule = \{(body(R), head(R)) \mid R \in \mathcal{R} \wedge body(R) \neq \emptyset\}$
- $Exc = \{(B, H) \mid \text{exception}(H, B) \in \mathcal{E}\}$
- $Existence = \{H \mid H \Leftarrow\in \mathcal{R}\}$
- $ExistenceSupport = \{(\{\text{ex}(H)\}, H) \mid H \in Existence\}$
- $Absence = Atom \backslash (\{head(R) \mid R \in \mathcal{R}\} \cup \{head(E) \mid E \in \mathcal{E}\})$
- $AbsenceAttack = \{(\text{ab}(B), B) \mid B \in Absence\}$
- $AR = Atom \cup \{\text{ex}(H) \mid H \in Existence\} \cup \{\text{ab}(B) \mid B \in Absence\}$
- $ATT = Exc \cup AbsenceAttack$
- $SUP = Rule \cup ExistenceSupport$

The following theorem indicates that the semantics is preserved during transformation [11].

Theorem 2. *For PROLEG program* P*, let* M *be an answer set of* P*. Assume that* $\mathcal{L}$ *is the complete labeling of the BAF transformed from* P*. Then, for each atom* H *in* P*,* $H \in M$ *iff* $\mathcal{L}(H) = in$*.*

Example 5. The program in Example 1 is transformed into the following BAF:
$\langle\ \{p, q_1, q_2, r, ex(q_2), ex(r)\},\ \{(r, q_1)\},$
$\{(\{q_1, q_2\}, p), (\{ex(q_2)\}, q_2), (\{ex(r)\}, r)\}\quad \rangle.$

Complete labeling of the BAF is performed in the following manner. Arguments q_1 and q_2 together support the argument p. The existence arguments $ex(q_2)$ and $ex(r)$ are added to support q_2 and r, respectively. Figure 4 shows a graphical representation of the BAF, with the complete labeling. As $\mathcal{L}(q_1) = out$ and $\mathcal{L}(q_2) = in$, the label of the set of arguments $\mathcal{L}(\{q_1, q_2\}) = out$. Also, as p is supported by $\{q_1, q_2\}$, $\mathcal{L}(p) = out$. When we ignore the existence and absence arguments introduced during transformation, the set of arguments labeled *in* is $\{q_2, r\}$, which coincides with the answer set of the program of Example 1.

5 Reasoning by the BAF

We describe the two types of reasoning performed by the BAF transformed from the PROLEG program:

1. A portrayal of the entire structure of judgment and the causality of arguments.
2. Identification of the required evidence.

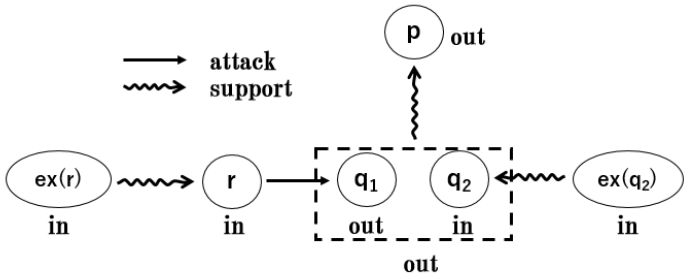


Fig. 4. BAF for the program in Example 1.

5.1 PROLEG program

Example 6. Consider the following PROLEG program. The first set of rules and exceptions states that if the object is a human (not a dead body) and there exists both the action of murder and also the intention to murder, then the crime of murder has been committed unless there was a legitimate defense. The second set of rules and exceptions states that if the accused is attacked and takes emergency, necessary, and appropriate action to defend himself/herself, then this is a legitimate defense, unless there was no intention to harm the deceased. The remainder of the program deals with the facts in evidence.

```
% rules regarding crime_of_murder
crime_of_murder <= human, act_of_murder, intention_to_murder.
exception(crime_of_murder, legitimate_defense).

legitimate_defense <=
  infringement, emergency, necessity, appropriateness,
  defense_intention.
exception(legitimate_defense, aggressive_intention_to_harm).

% facts
human <=.
act_of_murder <=.
intention_to_murder <=.
infringement <=.
emergency <=.
necessity <=.
appropriateness <=.
defense_intention <=.
```

5.2 The entire structure of judgment and causality of the arguments

In this case, the entire PROLEG program is transformed into a BAF using the rules shown in Section 4.

For each atom, a general rule defining both the atom and the exceptions is transformed into the BAF using a transformation rule. If there exists a fact, then a corresponding existence argument supporting the fact is added. If an atom does not appear in the header of any rule or exception, then a corresponding absence argument attacking the atom is added to the transformed BAF. We show a graphical representation of the transformed BAF in Figure 5.

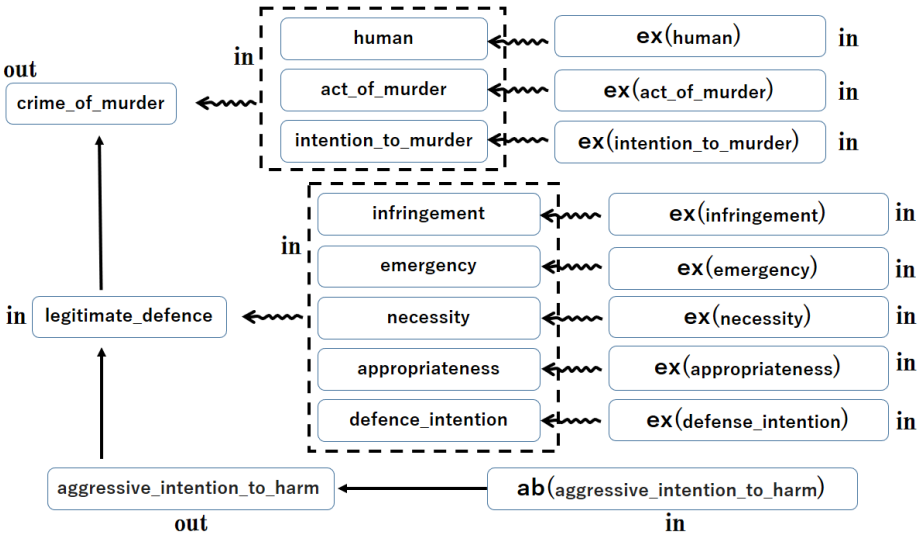


Fig. 5. Graphical representation of a transformed BAF for a murder case.

This BAF was obtained from an entire PROLEG program including facts, and shows the structure of the entire argumentation from which we can grasp the cause-and-effect relationships of the arguments.

Using this BAF, the argumentation process is explained as follows. As the label of the absence argument ab(aggressive_intention_to_harm) is *in*, that of the argument aggressive_intention_to_harm is *out* (there was no intention to harm). Therefore, the label of the argument legitimate_defense is *in* (it is a legitimate defense). The argument crime_of_murder has one attacking argument, the label of which is *in*, and one supporting set of arguments, the label of which is *in*. Hence, the label of the argument crime_of_murder is *out* (the crime of murder was not committed).

The BAF is updated as the judgment proceeds. Counter-arguments and evidences may be incrementally added as the corresponding nodes. Then, the

node labels can be changed. For example, if there is an exception to a legitimate_defense argument, and this is proven, a new argument is added; legitimate_defense is attacked by this argument and its label is changed to *out*. As another example, if evidence of aggressive_intention_to_harm is given, then its absence argument is replaced by an existence argument, and attack by the absence argument is replaced by support from the existence argument. As a result, the label of the node aggressive_intention_to_harm is changed to *in*. It follows that the label of legitimate_defense is changed to *out*, and that of the crime_of_murder to *in*.

5.3 Identification of required evidence

The BAF also identifies the evidence required to apply the law or prevent its application.

We transform a PROLEG program except for the fact part, and determine the existence arguments required to apply or not apply the law. Unlike the first type of reasoning, all available rules and exceptions are assumed to be represented, and no rules or exceptions are added.

From the definition of complete labeling, $\mathcal{L}(A) = in$ holds iff the labels of all arguments that attack A are *out* and there exists an argument that supports A, of which the label is *in*, or A is neither attacked nor supported.

Assume that a defendant wants to apply a law or that a plaintiff wants to prevent its application. Then they seek to label the corresponding argument *in* and *out*, respectively. The BAF determines the evidence required for attainment of either goal. This is achieved by repeatedly applying the following process:

Let A be an argument.

- Make $\mathcal{L}(A) = in$.
  Both of the following conditions should be satisfied.
  - (attack condition) Make $\mathcal{L}(B) = out$ for each B such that $\mathrm{att}(B, A)$. If there does not exist such an argument B, then the condition is satisfied.
  - (support condition) Make $\mathcal{L}(\mathbf{A}) = in$ for some $\mathbf{A}$ such that $\mathrm{sup}(\mathbf{A}, A)$, that is, for each $A' \in \mathbf{A}, \mathcal{L}(A') = in$. If there does not exist such $\mathbf{A}$, then an existence argument $ex(A)$ and a support $\mathrm{sup}(\{ex(A)\}, A)$ should be added.
- Make $\mathcal{L}(A) = out$.
  Either of the following conditions should be satisfied.
  - (attack condition) Make $\mathcal{L}(B) = in$ for some B such that $\mathrm{att}(B, A)$. If there does not exist such an argument B, then this condition is not satisfied.
  - (support condition) Make $\mathcal{L}(\mathbf{A}) = out$ for each $\mathbf{A}$ such that $\mathrm{sup}(\mathbf{A}, A)$, that is, for some $A' \in \mathbf{A}, \mathcal{L}(A') = out$. If there does not exist such $\mathbf{A}$, then this condition is not satisfied.

As a result, a set of existence arguments, that is, a set of evidences that should be provided, is found; this allows either party to attain his/her goal no matter what evidence his/her opponent offers.

Example 7. Figure 6 shows a PROLEG program excluding the fact part of Example 6. For convenience, each node is named $a, b, \ldots, k$, respectively.

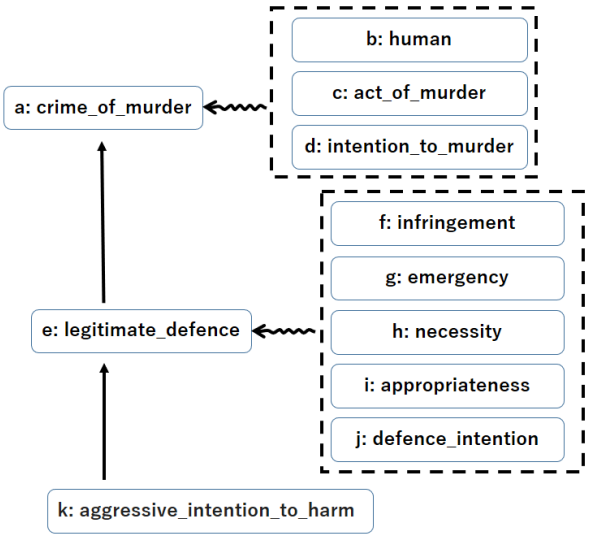


Fig. 6. Reasoning about required evidences.

– In this BAF, consider the conditions required to make $\mathcal{L}(a) = in$.
  - By attack condition for a, $\mathcal{L}(e) = out$ should be satisfied. To achieve this, attack condition for e or support condition for e should be satisfied. By attack condition for e, $\mathcal{L}(k) = in$ should be satisfied, and since k has no support, $ex(k)$ is required. By support condition for e, at least one of $\mathcal{L}(f) = out$, $\mathcal{L}(g) = out$, $\mathcal{L}(h) = out$, $\mathcal{L}(i) = out$ or $\mathcal{L}(j) = out$ holds. However, this is impossible since f, g, h, i and j are neither attacked nor supported.
  - By support condition for a, $\mathcal{L}(b) = \mathcal{L}(c) = \mathcal{L}(d) = in$ should be satisfied. Therefore, $ex(b), ex(c)$ and $ex(d)$ are required.

  As a result, the defendant should provide the four evidences $ex(k), ex(b), ex(c)$ and $ex(d)$ to apply the law.
– On the other hand, consider the conditions required to make $\mathcal{L}(a) = out$.
  - By attack condition for a, $\mathcal{L}(e) = in$ should be satisfied. To achieve this, $\mathcal{L}(k) = out$ should be satisfied, but this is impossible since k is neither attacked nor supported.
  - By support condition for a, either $\mathcal{L}(b)$, $\mathcal{L}(c)$ or $\mathcal{L}(d)$ should be out, but this is impossible since b, c and d are neither attacked nor supported.

  Therefore, the plaintiff never prevents application of the law.

In this example, only one set of existence arguments is found to make $\mathcal{L}(a) = in$, and no argument is found $\mathcal{L}(a) = out$. However, in general, we may find multiple sets in both cases. For example, assume that a defendant wishes to make $\mathcal{L}(a) = in$ in the BAF shown in Figure 7. The support required to make $\mathcal{L}(a) = in$ is one of $ex(b)$ or $ex(c)$. The support required to make $\mathcal{L}(f) = in$ is one of $ex(g)$ or $ex(h)$. Thus, we find four sets of required evidences.

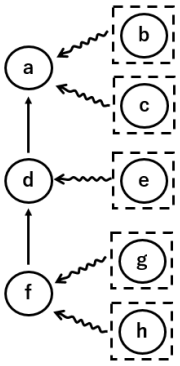


Fig. 7. Multiple sets of evidences are obtained.

6 Related Works

We compare our BAF semantics to the semantics derived by others.

Several works on BAF semantics have been undertaken. In almost all of them, the BAFs are given in advance or obtained by a translation from artificial logic programs. Such programs principally discuss argumentation structures that are seldom seen in actual judgments. We sought to apply real-world legal reasoning. A significant issue during transformation is to create BAF semantics preserving legal reasoning; no previous BAF semantics met this criterion.

Cayrol et al. investigated BAF semantics, defining several types of indirect attacks by combining attacks with supports. They also defined several types of extension [6]. Next, the concept of "coalition" (a set of arguments) was introduced and used to define a meta-AF [7, 8]. The idea was to reduce a BAF to an AF by deleting the support relations between arguments in the same coalition. An argument in BAF is accepted if it is included in an accepted coalition of the meta-AF. Boella et al. pointed out that this approach does not allow use of the Dung semantics, and revised the semantics by introducing different meta-arguments and meta-supports [2]. However, if we adopt these semantics, the semantics of PROLEG and BAF do not coincide [11]. It follows that we cannot

combine arguments to form a single support without considering their original relationships in PROLEG.

Noueioua et al. proposed a BAF that considered a support relation to be a "necessity" relation [12]. In this approach, each atom corresponds to each argument, similar to our approach. They proved the correspondence between a normal logic program and their BAF. The main drawback of the method is that it does not discriminate support by a set of arguments from support given by separate multiple arguments. They do not reflect the case in which a set of body goals support its head goal in a logic program.

Oren and Norman developed an evidential argumentation by introducing a special argument, corresponding to an environment, into a BAF [13]. This concept is similar to the existence argument of our method. The difference is that they introduced a single argument from which both attack and support relations arise. In contrast, we add an existence or absence argument for each fact. The structure of their evidential argumentation is more compact than ours, but the evidences bearing upon individual facts are unclear.

Unlike the works cited above, Brewka et al. developed an abstract dialectical framework (ADF) as a generalization of the Dung AF [3, 4]. In the ADF, each node is associated with an acceptance condition depending on the parent nodes, and each link exhibits an individual strength. A bipolar ADF is a subclass of ADF in which a link is either attacked or supported depending on the polarity of its strength. A BAF transformed from PROLEG may be considered to be an instantiation of an ADF. It would be interesting to explore whether an ADF semantics could be simply applied to a BAF transformed from PROLEG.

7 Conclusion

We have described the transformation from a PROLEG description to a BAF, and the legal reasoning using the BAF. We gave semantics to the BAF preserving the features of a PROLEG program. The BAF reflects the structure of the judgment process and causality among arguments. We have developed reasoning on the BAF, that is difficult to emulate or understand using a PROLEG program or trace of its execution. Our system enables lawyers and law school students to analyze judgments.

In future, we will improve reasoning by the BAF and create a graphical interface.

Acknowledgment

This work was supported by JSPS KAKENHI Grant Number JP17H06103.

References

1. Bench-Capon, T., Prakken H. and Sartor, G.: Argumentation in legal reasoning. *Argumentation in Artificial Intelligence*, 363-382 (2009).

2. Boella, G., Gabbay, D. M., Torre, L. van der and Villata, S.: Support in abstract argumentation In *Proc. of COMMA2010*, 40-51 (2010).
3. Brewka, G. and Woltran, S.: Abstract dialectical frameworks. In *Proc. of KR2010*,102-111 (2010).
4. Brewka, G., Ellmauthaler, S., Strass, H., Wallner, J.P. and Woltran, S.: Abstract dialectical frameworks revisited. In *Proc. of IJCAI2013*, 803-809 (2013).
5. Caminada, M.: On the issue of reinstatement in argumentation. In *Proc. of JELIA2006*, 111-123 (2006).
6. Cayrol, C. and Lagasquie-Schiex, M.: On the acceptability of arguments in bipolar argumentation frameworks. In *Proc. of ECSQARU2005*, 378-389 (2005).
7. Cayrol, C. and Lagasquie-Schiex, M.: Coalitions of arguments: A tool for handling bipolar argumentation frameworks. In *International Journal of Intelligent Systems*, Volume 25, 83-109 (2010).
8. Cayrol, C. and Lagasquie-Schiex, M.: Bipolarity in argumentation graphs: Towards a better understanding. *International Journal of Approximate Reasoning*, Volume 54, 876-899 (2013).
9. Dung, P. M.: On the acceptability of arguments and its fundamental role in non-monotonic reasoning, logic programming and n-person games. *Artificial Intelligence*, Volume 77, 321-357 (1995).
10. Gelfond, M. and Lifschitz, V.: The stable model semantics for logic programming. In *Proc. of ICLP*, 1070-1080 (1988).
11. Kawasaki, T., Moriguchi, S. and Takahashi, K.: Transformation from PROLEG to a bipolar argumentation framework. In *Proc. of SAFA2018*, 36-47 (2018).
12. Nouioua, F. and Risch, V.: Argumentation framework with necessities. In *Proc. of SUM2011*, 163-176 (2011).
13. Oren, N. and Norman, T. J.: Semantics for evidence-based argumentation. In *Proc. of COMMA2008*, 276-284 (2008).
14. Prakken, H., Reed, C. and Walton, D.: Dialogues about the burden of proof. In *Proc. of ICAIL2005*, 115-124 (2005).
15. Rahwan, I. and Simari, G.(eds.): *Argumentation in Artificial Intelligence*, Springer (2009).
16. Reed, C. and Rowe, G.: Araucaria: Software for argument analysis, diagramming and representation. In *International Journal of AI Tools*, Volume 13, 961-980 (2004).
17. Satoh, K. et al.: PROLEG: An Implementation of the Presupposed Ultimate Fact Theory of Japanese Civil Code by PROLOG Technology. In *JSAI-isAI 2010: New Frontiers in Artificial Intelligence*, 153-164 (2010).
18. Satoh, K. et al.: On generality of PROLEG knowledge representation. In *Proc. of JURISIN2012*, 115-128 (2012).

An empirical evaluation of AMR parsing for legal documents

Vu Trong Sinh and Nguyen Le Minh

Japan Advanced Institute of Science and Technology (JAIST)
sinhvtr@jaist.ac.jp, nguyenml@jaist.ac.jp

Abstract. Many approaches have been proposed to tackle the problem of Abstract Meaning Representation (AMR) parsing, helps solving various natural language processing issues recently. In our paper, we provide an overview of different methods in AMR parsing and their performances when analyzing legal documents. We conduct experiments of different AMR parsers on our annotated dataset extracted from the English version of Japanese Civil Code. Our results show the limitations as well as open a room for improvements of current parsing techniques when applying in this complicated domain.

Keywords: abstract meaning representation · semantic parsing · legal text.

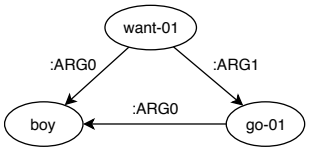
1 Introduction

In Natural Language Processing, semantic representation of text plays an important role and receives growing attention in the past few years. Many semantic schemes have been proposed, such as Groningen Meaning Bank [1], Abstract Meaning Representation [13], Universal Conceptual Cognitive Annotation [17]. In which, Abstract Meaning Representation (AMR) has shown a great potential and gained popularity in computational linguistics [25] [8] [10] [12].

AMR is a semantic representation language that encodes the meaning of a sentence as a rooted, directed, edge-labeled, leaf-labeled graph while abstracting away the surface forms in a sentence. Every vertex and edge of the graph are labeled according to the sense of the words in a sentence. AMR can be represented in PENMAN notation, for a human to read and write easily, or graph structure, for a computer to store in its memory, or decomposed into conjunctions of logical triples, for calculating the difference among AMRs. Table 1 shows an example of AMR annotation for the sentence *"The boy wants to go"* with different formats mentioned above.

AMR has been applied as an intermediate meaning representation for solving various problems in NLP including machine translation [11], summarization [15], event extraction [27], [21], [14], machine comprehension [20]. For AMR to be useful in these problems, the AMR parsing task, which aims to map a natural language string to an AMR graph, plays a crucial role. Despite the advantages in handling semantic attributes of text, there are not many works exploring the

Table 1: Abstract meaning representation for the sentence *"The boy wants to go"* in three formats

$\exists w, b, g : instance(w, want-01) \wedge instance(g, go-01) \wedge instance(b, boy) \wedge arg0(w, b) \wedge arg1(w, g) \wedge arg0(g, b)$	(w / want-01 :arg0 (b /boy) :arg1(g / go-01) :arg0 b)	

application of AMR in analyzing legal documents. Unlike other domains, understanding legal text faces a number of challenges due to the special characteristics such as complicated structures, long sentences, domain-specific terminology.

In this paper, we would like to investigate the potential of AMR in this interesting field. We provide an overview of main approaches in current AMR parsing techniques in section 2. From each approach, we choose best systems that already published the source code to conduct our experiments. In section 3, we revise the dataset **JCivilCode-1.0** introduced in 2017 by Lai et al. [22] with some modifications and additional samples. We also extract sentences with various lengths from a well-known dataset LDC2017T10 [1] in common domain to have more observation on the performances of each system. Our results and some discussions are provided in section 4.

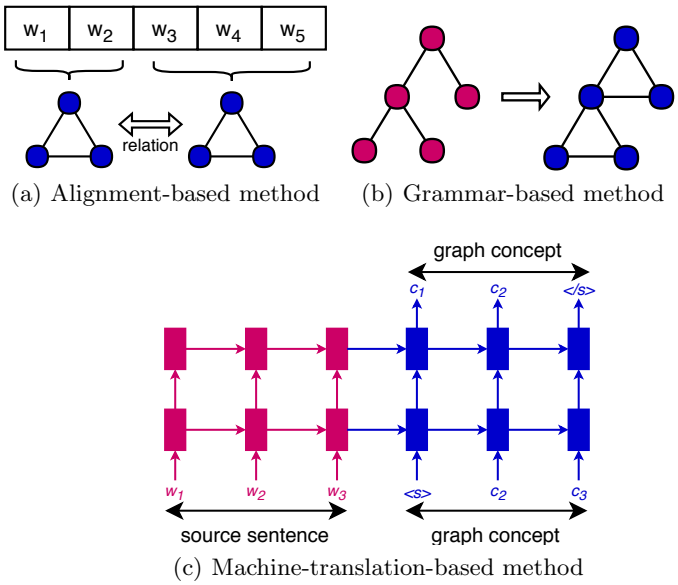
2 Approaches in AMR parsing

2.1 AMR notation

In AMRs, each node is named by an ID (variable). It contains the semantic concept, which can be a word (e.g. *boy*) or a PropBank frameset (e.g. *want-01*) or a special keyword. The keywords consist of entity type (e.g. *date-entity, ordinal-entity, percentage-entity*), quantities (e.g. *distance-quantity*), and logical conjunction(e.g. *and, or*). The edge between two vertices is labeled using more than 100 relations including frameset argument index (e.g. *:ARG0, :ARG1*), semantic relations (e.g. *:location, :name*), relations for quantities (e.g. *:quant, :unit, :scale*), relations for date-entities, relations for listing (e.g. *:op1, :op2, :op3*). AMR also provides the inverse form of all relations by concatenating -of to the original relation (e.g. :location vs :location-of). Hence, if r is a directed relation of two entities a and b, we have $R(a, b) \equiv R - of(b, a)$. This inverse relation helps keep the focus on the entity instead of the verb sense as default.

The task of parsing a natural language text into an AMR graph faces a lot of challenges, such as word-sense disambiguation, semantic graph construction, data sparsity. Many approaches have been proposed to tackle this problem. They can be divided into three main categories: alignment-based, grammar-based and machine-translation-based.

[1] https://catalog.ldc.upenn.edu/ldc2017t10



(a) Alignment-based method (b) Grammar-based method

(c) Machine-translation-based method

Fig. 1: Main approaches in AMR parsing

2.2 Alignment-based parsing

One of the pioneer AMR parsing solutions is **JAMR** introduced by Flanigan et al. in 2014 [7], which build a two-part algorithm that first identifies concepts with an automatic aligner and then identifies the relations that it obtains between these by searching for the maximum spanning connected subgraph from an edge-labeled, directed graph representing all possible relations between the identified concepts. This method provided a strong baseline in AMR parsing. Follow this approach, Zhou et al. [28] extended the relation identification tasks with a component-wise beam search algorithm. Chunchuan and Titov [5] improved this method by considering alignments as latent variables in a joint probabilistic model. They used variational autoencoding technique to perform the alignment inference and archieved the state-of-the-art in AMR parsing until now. But the source code for this model has not been published completely yet. In this paper, we take the JAMR model [7] to analyze and conduct experiments.

The core idea of alignment-based methods is to construct a concept set by aligning the Propbank concepts with the words that evoke them. The authors build an automatic aligner that uses a set of rules to greedily align concepts to words. The authors use WordNet to generate candidate lemmas and a fuzzy match of a concept, defined to be a word in the sentence that has the longest string prefix match with that concept's label. For instance, the fuzzy match for **apply-01** could be aligned with *"application"* if this word is the best match in

the sentence. Figure 2 shows an example of aligning words in a sentence with AMR concepts. This JAMR aligner is widely used in many later works.

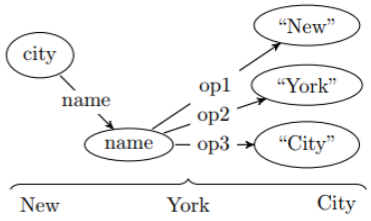


Fig. 2: Alignment of the words span "New York City" with AMR fragment [7]

In the first stage of identifying concepts, given a sentence $w = \{w_1, w_2, ..., w_n\}$, the parser segments w into subsequences, denoted $\{w_{b_0:b_1}, w_{b_1:b_2}, ..., w_{b_{k-1}:b_k}\}$, called contiguous spans. A span $\{w_{b_{i-1}:b_i}\}$ is then assigned to a concept graph fragment c_i from the concept set $clex\{w_{b_{i-1}:b_i}\}$, or to θ for words that evoke no concept. This assigning between sequence of spans b and concept graph fragment c is calculated by a score function:

$$score(b, c; \theta) = \sum_{i=1}^{k} \theta^T f(w_{b_{i-1}:b_i}, b_{i-1}, c_i), \tag{1}$$

where f is a feature vector representation of a span and one of its concept graph fragments in the sentence. The features can be fragment given words, length of the matching span, name entity recognizing or bias.

To find the highest-scoring between b and c, JAMR uses a semi-Markov model. Let $S(i)$ be the score of the first i words of the sentence $(w_{o:i})$. Then $S(i)$ is calculated recurrently via the previous scores, with the initialization $S(0) = 0$. Obviously, $S(n)$ becomes the best score. To obtain the best scoring concept labeling, JAMR uses back-pointers method, similar to the implementation of the Viterbi algorithm [18].

The second stage is to identify the relation, which sets the edge label among the concept subgraph fragments assigned in the previous stage. The authors tackle this stage like a graph-based dependency parser problem. While the dependency parser aims to find the maximum-scoring tree over words from the sentence, the relation identifier tries to find the maximum-scoring among subgraphs that preserve concept fragments from the previous stage.

To train the two stage parser, the authors formulate the training data for concept identification and relation identification separately. In both tasks, the input must be annotated with name entities (obtained from Illinois Name Entity Tagger), part-of-speech tags and basic dependencies (obtained from Stanford Parser). The detail settings and hyper-parameters can be found in the original paper. This parser has been evaluated the first time on LDC2013E117 corpus (in

2014), archived the Smatch score of ***0.58***, and the second time on LDC2015E86 corpus (in 2016), which showed great improvement with ***0.67*** Smatch score.

2.3 Grammar-based parsing

After the success of Flanigan et al. with an alignment-based approach [7], Wang et al. [24] introduced a grammar-based (or transition-based) parser called **CAMR**. The authors first use a dependency parser to generate the dependency tree for the sentence, then transform the dependency tree to an AMR graph through some transition rules. This method takes advantages of the achievements in dependency parsing, with a training set much larger than the training set of AMR parsing. Damonte et al. [16]. Brandt et al. [2], Goodman et al. [9] and Peng et al. [26] also applied the grammar-based algorithm in their works and obtained competitive results. Figure 3 shows an example of the dependency tree and the AMR graph parsed from the same sentence *"Private rights must conform to the public welfare"*.

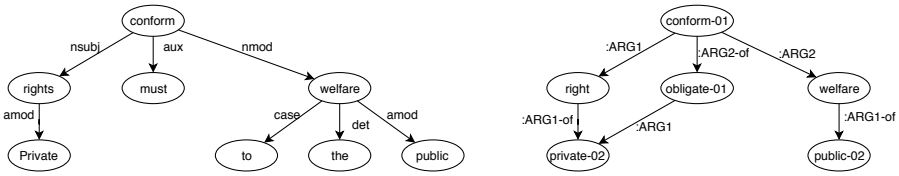


Fig. 3: Dependency tree and AMR graph generated from the sentence *"Private rights must conform to the public welfare"*

Unlike the dependency tree of a sentence, where each word corresponds to a node in the tree, in AMR graph, some words become abstract concepts or relations while other words are simply deleted because they do not contribute to the meaning. This difference causes many difficulties for aligning word tokens and the concept. In order to learn the transition from the dependency tree to AMR graph, Wang et al. [24] use the algorithm from JAMR to produce the alignment. The authors also construct a span graph to represent an AMR graph that is aligned with the word tokens in a sentence. This span graph is a directed, labeled graph $G = (V, A)$, where $V = \{s_{i,j}|i, j \in (0, n)$ and $j > i\ \}$ is a set of nodes, and $A \subseteq V \times V$ is a set of arcs. Each node $s_{i,j}$ in G is assigned a concept label from concept set L_V and is mapped with a continuous span $(w_i, ..., w_{j-1})$ in the sentence w. Each arc is also assigned a relation label from relation set L_A.

Basically, CAMR will perform three types of actions to transform the dependency tree into the AMR graph: actions performed when an edge is visited, actions performed when a node is visited, and actions used to infer abstract concepts in AMR that does not correspond to any word or word sequence in the sentence. For the details of these actions, readers can refer to the original paper [23], the Boosting version [4] and the paper at Semeval2016 contest [24].

A disadvantage of this method is that it limits the parsing ability to a single sentence, because the dependency tree can cover only the structure inside a sentence.

Damonte et al. [16] developed a transition-based model called **AMREager** that also parses the AMR graph based on transition rules, but differs from CAMR which requires the full dependency tree to be obtained and then process the tree bottom-up, this parser process the sentence left-to-right. AMREager defines a stack, a buffer and a configuration to perform the transition actions, which can be: *Shift, LArc, RArc* or *Reduce*. AMREager also uses the alignment obtained from JAMR aligner to map indices from the sentence to AMR graph fragments. Although the result in Smatch score is still lower than CAMR and JAMR by a small margin, AMREager obtains best results on several subtasks such as Name Ent. and Negation.

2.4 Machine-translation-based parsing

Recently, with the achievement of the encoder-decoder architecture in deep neural networks, several supervised learning approaches have been proposed in order to deal with AMR parsing task. They attempt to linearize the AMR in Penman notation to sequences of text, at character-level [19] or at word-level [12] [8], so that the parsing task can be considered as a translation task, which transforms a sentence into an AMR-like sequence. In this paper, we choose **NeuralAMR** (word-level linearization) [12] and **Ch-AMR** (character-level linearization) [19] to run our experiments.

Table 2: AMR linearization for the phrase *"US officials"* in NeuralAMR - the left side is the original AMR and the right side is the linearized string

(o / obligate-01 :ARG1 (r / right-05 :ARG1-of (p / private-02)) :ARG2 (c / conform-01 :ARG1 r :ARG2 (w / welfare :ARG1-of (p2 / public-02))))	(obligate-01 :ARG1 (right-05 :ARG1-of (private-02)) :ARG2 (conform-01 :ARG1 (right-05) :ARG2 (welfare :ARG1-of (public-02))))

Given an AMR graph represented in Penman notation, NeuralAMR preprocesses the graph through a series of steps: AMR linearization, anonymization, and other modifications which aim to reduce the complexity of the linearized sequences and to address sparsity from certain open class vocabulary entries, such as named entities and quantities. Representing AMR graphs in this way, NeuralAMR takes advantage of sequence-to-sequence model by using a stack bidirectional LSTM encoder to encode the input sequence and a stacked LSTM

to decode from the hidden states produced by the encoder. The output string of the model is converted back to AMR format to complete the parsing process. Since this approach requires a huge amount of labeled data, NeuralAMR uses paired training procedure to bootstrap a high-quality AMR parser from millions of unlabeled Gigaword sentences. With this extra dataset, the parsing result increases significantly, from ***0.55*** to ***0.62*** in Smatch score. However, it is difficult for this linearization method to keep the structure of the original graph. In the example shown in Figure 1(b), the distance between two nodes: *"obligate-01"* and *"conform-01"*, which are directly connected in the graph, becomes 5 (tokens) in the linearized string, as shown in Table 2. This distance can be even larger in long sentences with complicated structure, thus causes many mistakes in the annotation.

P r i v a t e ***JJ*** r i g h t s ***NNS*** m u s t ***MD*** c o n f o r m ***VB*** t o ***TO*** t h e ***DT*** p u b l i c ***NN*** w e l f a r e ***NN***
(obligate-01 :ARG1 (right-05 :ARG1-of (private-02)) :ARG2 (conform-01 :ARG1 (right-05) :ARG2 (welfare :ARG1-of (public-02))))

Fig. 4: Preprocessing data in Ch-AMR - The sentence is converted to sequence of characters with POS tag in uppercase following each word, the AMR graph is linearized and removed all variables

Different from Kontas et al. [12], Noord and Bos [19] introduce another approach in linearizing which transforms the AMR graph to the character-level. This model removes all variables from the AMRs and duplicate co-referring nodes. The input sentences are also tokenized in character-level, along with the part-of-speech tag of the original tokens to provide more linguistic information to the decoder. An example of such a preprocessed AMR is shown in Figure 4 Obviously, this preprocessing method causes losing information, since the variables cannot be put back perfectly. To tackle this limitation, the authors describe an approach to restore the co-referring nodes in the output. All wikification relations present in AMRs in the training set are also removed and restored in a post-processing step. This model archives a better result, with **0.71** Smatch score.

3 Dataset and evaluation

3.1 Dataset

The original dataset used for testing in this paper is JCivilCode-1.0, which is introduced by Lai et al. in [22]. In our work, we revised JCivilCode-1.0 carefully with some modifications and extracted more 48 articles to complete the first four

chapter in Part I of the Japanese Civil Code. All the AMRs are annotated by a group of annotators to ensure the neutrality of evaluation. Table 3 shows some statistics of this dataset after our revision.

As we mentioned in section 1, one of the main difficulty in analyzing legal documents is dealing with long sentences. In our experiments, we also would like to assess the performances of the five models with different length of the sentence. Since the current legal dataset is still small, we use extra sentences extracted from the well-known LDC2017T10 dataset, which consists of nearly 40,000 sentences in news domain. We divide the test set of LDC2017T10 into four subsets LDC-20, LDC-20-30, LDC-30-40, LDC-40 with the lengths of the sentences in range 0-20, 21-30, 31-40 and greater than 40 words, respectively. We excluded the samples containing sub-sentences inside (annotated *"multi-sentence"* by the annotators). This exclusion guaranteed a fair comparison among the five parsers because CAMR is unable to analyze multiple sentences at the same time.

Table 3: JCivilCode-1.0 statistic

Number of samples	128
Average length	31
Max sentence length	107
Average number of graph nodes	28
Max number of graph nodes	96
Vocabulary size	796
Number of tokens	4042

3.2 Evaluation

AMR parsers are evaluated mainly by Smatch score [3]. Given the parsed graphs and the gold graphs in the form of Penman annotations, Smatch first tries to find the best alignments between the variable names for each pair of graphs and it then computes precision, recall and F1 of the concepts and relations. In this paper, to test the performance of AMR parser on legal text, which contains sentences in complicated structures, we analyze the parsing results in a deeper measurement. Specifically, we use the test-suite introduced by Damonte et al. [16], which assesses the parsing results on various sub-score as follow:

- *Unlabeled*: Smatch score computed on the predicted graphs after removing all edge labels (e.g., *:ARG0, :condition*)
- *No WSD*: Smatch score while ignoring Propbank senses (e.g., *perform-01* vs *perform-02*)
- *Name Entity*: F-score on the named entity recognition (:name roles)
- *Wikification*: F-score on the wikification (:wiki roles)
- *Negation*: F-score on the negation detection (:polarity roles)

- *Concepts*: F-score on the concept identification task
- *Reentrancies*: Smatch computed on reentrant edges only
- *SRL*: Smatch computed on :ARG-i roles only

In our experiment with JCivilcode-1.0, we do not include the Wikification and Name Entity criteria since there are no Wiki concepts included in this dataset, and the number of existing named entities is small.

4 Experiments and discussion

To evaluate the performance of different parsing strategies on legal text, we conduct experiments on five models that already provided their source codes: JAMR, CAMR, AMR-Eager, NeuralAMR and Ch-AMR. While JAMR, CAMR and AMR-Eager were trained with the LDC2015E86 dataset only (the older version of LDC2017T10), NeuralAMR and Ch-AMR initialized the parser by LDC2015E86 and then used an extra corpus of 2 millions sentences extracted from a free text corpus Gigaword [6] to train the complete models. We provide some statistics about LDC2015E86 and LDC2017T10 in Table 4. English sentences in these two datasets are collected from TV program transcriptions, web blogs and forums. Each sample in these datasets includes a pair of sentence and AMR graph corresponding.

Table 4: LDC2015E86 and LDC2017T10 number of samples

Dataset	Total	Train	Dev	Test
LDC2015E86	19,572	16,833	1,368	1,371
LDC2017T10	39,260	36,521	1,368	1,371
LDC-20	-	-	-	694
LDC-20-30	-	-	-	284
LDC-30-40	-	-	-	143
LDC-40	-	-	-	82

Parsing results are summarized in Table 5. The first row is the Smatch score of five models on LDC2015E86 dataset, reported in their original papers. The four next rows are the Smatch score on LDC2017T10 test set with different ranges of sentence length and the remaining rows show the Smatch score and sub-scores on JCivilCode-1.0.

Overall, the Smatch score of all the parsers on JCivilCode-1.0 is still lower than on LDC2015E86 by a large margin. It can be figured out that grammar-based and alignment-based methods showed promising results over MT-based method. JAMR and CAMR archieved the best score on LDC2017T10 long sentences and JCivilCode-1.0 dataset, respectively, while AMREager's performance was competitive on both tasks.

In LDC2017T10 experiments, JAMR remained the best parser in every range of sentence length. The gap between this method and the others even becomes

Table 5: Smatch scores and sub-scores of five models

	JAMR	**CAMR**	**AMREager**	**NeuralAMR**	**Ch-AMR**
LDC2015E86	**0.67**	0.66	0.64	0.62	0.71
LDC-20	**0.71**	0.66	0.69	0.65	0.45
LDC-20-30	**0.68**	0.62	0.64	0.59	0.43
LDC-30-40	**0.66**	0.60	0.62	0.56	0.42
LDC-40	**0.65**	0.59	0.62	0.54	0.40
JCivilCode-1.0	0.45	**0.48**	0.43	0.39	0.28
Unlabeled	0.50	**0.56**	0.53	0.46	0.37
No WSD	0.47	**0.50**	0.45	0.40	0.28
Negation	0.23	0.16	0.32	**0.35**	0.19
Concepts	0.59	**0.63**	0.62	0.52	0.35
Reentrancies	0.32	**0.35**	0.31	0.29	0.22
SRL	0.43	**0.47**	0.41	0.00	0.28

larger when parsing longer sentences. Although grammar-based methods focus on constructing the structure of the graph based on its corresponding dependency tree, CAMR and AMREager are unable to provide better output than JAMR.

In legal text parsing experiments, CAMR outperforms the others on both the Smatch score and many sub-scores. Specifically, this method obtains best results in constructing graph topology (*Unlabeled*), predicting the Propbank sense (No WSD and SRL) as well as identifying concepts in AMR graphs (Concepts score). When parsing graphs containing cycles, CAMR also performs better, as shown in Reentrancies scores.

We analyze some common errors in parsing outputs of legal sentences, with the statistic represented in Figure 5 and the examples provided in Table 6. One of the most common error in alignment-based and grammar-based performances is missing concept and relation related to *modal verbs*. In legal documents, modal verbs (e.g. "may", "can", "must") play a crucial role in a sentence and decide whether an action is permitted or not. This differs from other domains, where these words do not often contribute a lot to the sentence meaning. As shown in example 1 in Table 6, only NeuralAMR is capable of identifying the concept *"obligate-01"* while other models totally ignore it.

Another challenge in parsing legal text is the logical complexity. In this aspect, all the parsers still show limitation when parsing negative clause. This is not too surprising as many negations are encoded with morphology (e.g., such as *un-* prefix in *"unless"* or *"unable"*) and cause difficulties for detection. We show an example in Table 6 the output from all the parsers for a sentence: *"No abuse of rights is permitted"*. NeuralAMR and JAMR succeeded in converting negation to *:polarity* -, AMREager didn't put this edge to the exact position, but in this case, it doesn't change the meaning of the sentence. CAMR even performs worse as it skips this important information.

Table 6: Common errors types: **Incorrect concept** - **Incorrect relation** - **Missing concept** - **Missing attribute**

Example	*Private rights must conform to the public welfare* (1)	*No abuse of rights is permitted* (2)
Gold annotation	(o / obligate-01 :ARG1 (r / right-05 :ARG1-of (p / private-02)) :ARG2 (c / conform-01 :ARG1 r :ARG2 (w / welfare :ARG1-of (p2 / public-02))))	(p / permit-01 :polarity - :ARG1 (a / abuse-01 :ARG1 (r / right-05)))
JAMR	(c / conform-01 :ARG1 (r / **right** :ARG1-of (p2 / **private-03**)) :ARG2 (w / welfare **:domain-of** (p / **public**))) (missing concept **obligate-01**)	(p / permit-01 :ARG1 (a / abuse-01 :ARG1 (r / **right**)) :polarity -)
CAMR	(x2 / right-05 :ARG1-of (x1 / **private-03**) :ARG1-of (x4 / conform-01 :ARG2 (x8 / welfare **:mod** (x7 / **public**)))) (missing concept **obligate-01**)	(x6 / permit-01 :ARG1 (x2 / abuse-01 :ARG1 (x4 / **right**))) (missing attribute **:polarity -**)
AMR-Eager	(v3 / conform-01 :ARG1 (v2 / **right** :ARG1-of (v1 / **private-03**)) :ARG2 (v5 / welfare **:mod** (v4 / **public**))) (missing concept **obligate-01**)	(v3 / permit-01 :ARG1 (v1 / abuse-01 polarity - :ARG1 (v2 / **right**)))
Neural-AMR	(o / obligate-01 :arg2 (r / **rule-out-02** :arg0 (r2 / **right** :arg1-of (p / **private-03**)) :arg1 (w / welfare :mod (p2 / public))))	(p / permit-01 :polarity - :arg1 (a / **abuse-02** :arg1 (r / **right**)))
Ch-AMR	(vv1conform-01 / conform-01 :ARG1 (vv1person / **person** :ARG1-of (vv1private-03 / **private-03**)) :ARG2 (vv1welfare / welfare :ARG1-of **vv1**)) (missing concept **"right-05", "public-02", "obligate-01"**)	(vv3permit-01 / permit-01 :ARG1 (vv3no-abuse / **no-abuse**)) (missing concept **"right-05"**)

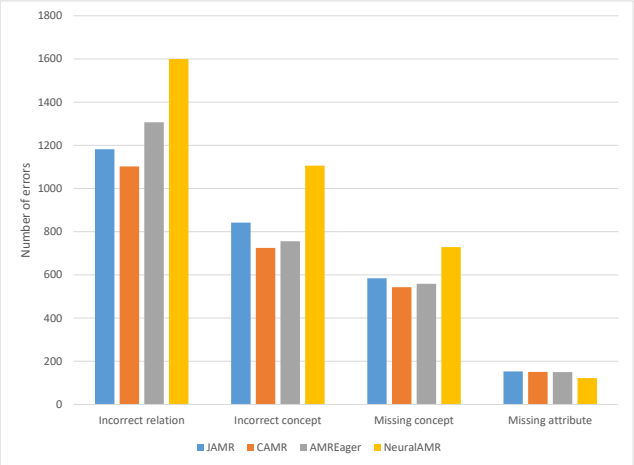


Fig. 5: Common error types

5 Conclusions

We conducted experiments of AMR parsing on the legal dataset JCivilCode-1.0 and news domain dataset LDC2017T10 with different ranges of sentence length to observe the abilities of five different models. The parsing outputs were evaluated by Smatch metric in several aspects including overall F-score and sub-score on specific tasks. Experimental results showed the domain adaptation of five models for the legal domain and the performance decreased of approximately 0.2 on the Smatch score. This result shows difficulties in applying AMR parsing for analyzing legal documents.

Currently, our legal dataset JCivilcode is still too small comparing to LDC2017T10. In order to improve the domain adaptation ability for current approaches as well as to obtain a fully evaluation, the legal dataset has to be enlarged. This work requires a lot of efforts from experts in both linguistic and legal domain.

Acknowledgment

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

References

1. Basile, V., Bos, J., Evang, K., Venhuizen, N.: Developing a large semantically annotated corpus. In: Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC 2012). pp. 3196–3200. Istanbul, Turkey (2012)
2. Brandt, L., Grimm, D., Zhou, M., Versley, Y.: ICL-HD at SemEval-2016 Task 8: Meaning Representation Parsing - Augmenting AMR Parsing with a Preposition Semantic Role Labeling Neural Network. In: Proceedings of the 10th International

Workshop on Semantic Evaluation (SemEval-2016). pp. 1160–1166. Association for Computational Linguistics (2016)
3. Cai Shu, Knight Kevin: Smatch: an Evaluation Metric for Semantic Feature Structures. In: Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics. pp. 748–752. Association for Computational Linguistics (2013)
4. Chuan Wang, Nianwen Xue, S.P.: Boosting Transition-based AMR Parsing with Refined Actions and Auxiliary Analyzers. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing. pp. 857–862. Association for Computational Linguistics (2015)
5. Chunchuan Lyu, Ivan Titov: AMR Parsing as Graph Prediction with Latent Alignment. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics. p. 397407 (2018)
6. Courtney Napoles, Matthew Gormley, Benjamin Van Durme: Annotated Gigaword. In: Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction. Association for Computational Linguistics (2012)
7. Flanigan, J., Thomson, S., Carbonell, J., Dyer, C., Smith, N.A.: A Discriminative Graph-Based Parser for the Abstract Meaning Representation (2014)
8. Gildea, D., Xue, N., Peng, X., Wang, C.: Addressing the Data Sparsity Issue in Neural AMR Parsing. In: EACL (2017)
9. Goodman, J., Vlachos, A., Naradowsky, J.: UCL+Sheffield at SemEval-2016 Task 8: Imitation learning for AMR parsing with an alpha-bound. In: SemEval@NAACL-HLT (2016)
10. Jeffrey Flanigan, Chris Dyer, Noah A. Smith, Jaime Carbonell: Generation from abstract meaning representation using tree transducers. In: Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics. pp. 731–739. Sandiego, California (2016)
11. Jones, B., Andreas, J., Bauer, D., Moritz Hermann, K., Knight, K.: Semantics-Based Machine Translation with Hyperedge Replacement Grammars. In: 24th International Conference on Computational Linguistics - Proceedings of COLING 2012: Technical Papers. pp. 1359–1376 (12 2012)
12. Konstas, I., Iyer, S., Yatskar, M., Choi, Y., Zettlemoyer, L.: Neural AMR: Sequence-to-Sequence Models for Parsing and Generation. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 146–157. Association for Computational Linguistics (2017). https://doi.org/10.18653/v1/P17-1014, http://www.aclweb.org/anthology/P17-1014
13. Laura Banarescu, Claire Bonial, Shu Cai, Madalina, Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, Nathan Schneider: Abstract Meaning Representation for Sembanking. In: Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse. pp. 178–186 (2013)
14. Lifu Huang, Taylor Cassidy, Xiaocheng Feng, Heng Ji, Clare R. Voss, Jiawei Han, Avirup Sil: Liberal Event Extraction and Event Schema Induction. In: ACL (2016)
15. Liu, F., Flanigan, J., Thomson, S., Sadeh, N., Smith, N.A.: Toward Abstractive Summarization Using Semantic Representations. In: NAACL. pp. 1077–1086 (2015)
16. Marco Damonte, Giorgio Satta and Shay B. Cohen: An Incremental Parser for Abstract Meaning Representation. In: EACL (2017)

17. Omri Abend, Ari Rappoport: Universal Conceptual Cognitive Annotation (UCCA). In: Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics. p. 228238 (2013)
18. Rabiner, L.R.: A tutorial on Hidden Markov Models and selected applications in speech recognition. Proceedings of the IEEE **77**(2), 257–286 (Feb 1989). https://doi.org/10.1109/5.18626
19. Rik van Noord, Johan Bos: Neural Semantic Parsing by Character-based Translation: Experiments with Abstract Meaning Representations. Computational Linguistics in the Netherlands Journal **7**, 93–108 (2017)
20. Sachan, M., Xing, E.: Machine Comprehension using Rich Semantic Representations. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers). pp. 486–492. Association for Computational Linguistics (2016). https://doi.org/10.18653/v1/P16-2079, http://www.aclweb.org/anthology/P16-2079
21. Sudha Rao, Daniel Marcu, Kevin Knight, Hal Daum: Biomedical Event Extraction using Abstract Meaning Representation. In: BioNLP (2017)
22. Viet, L.D., Sinh, V.T., Minh, N.L., Satoh, K.: Convamr: Abstract meaning representation parsing for legal document. CoRR **abs/1711.06141** (2017), http://arxiv.org/abs/1711.06141
23. Wang Chuan, Xue Nianwen, P.S.: A Transition-based Algorithm for AMR Parsing. In: Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. pp. 366–375. Association for Computational Linguistics (2015). https://doi.org/10.3115/v1/N15-1040, http://www.aclweb.org/anthology/N15-1040
24. Wang Chuan, Pradhan Sameer, Pan Xiaoman, Ji Heng, Xue Nianwen: CAMR at SemEval-2016 Task 8: An Extended Transition-based AMR Parser. In: Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016). pp. 1173–1178. Association for Computational Linguistics, San Diego, California (June 2016), http://www.aclweb.org/anthology/S16-1181
25. Wang Yanshan, Liu Sijia, Rastegar-Mojarad Majid, Wang Liwei, Shen Feichen, Liu Fei, Liu Hongfang: Dependency and AMR Embeddings for Drug-Drug Interaction Extraction from Biomedical Literature. In: Proceedings of the 8th ACM International Conference on Bioinformatics, Computational Biology,and Health Informatics. pp. 36–43. ACM-BCB '17, ACM, New York, NY, USA (2017). https://doi.org/10.1145/3107411.3107426, http://doi.acm.org/10.1145/3107411.3107426
26. Xiaochang Peng, Daniel Gildea, G.S.: AMR Parsing With Cache Transition Systems. In: AAAI (2018)
27. Xiaoman Pan, Taylor Cassidy, Ulf Hermjakob, Heng Ji, Kevin Knight: Unsupervised Entity Linking with Abstract Meaning Representation. In: Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. pp. 1130–1139 (2015). https://doi.org/10.3115/v1/N15-1119, http://www.aclweb.org/anthology/N15-1119
28. Zhou Junsheng, Xu Feiyu, U.H.Q.W.L.R.G.Y.: AMR Parsing with an Incremental Joint Model. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing. pp. 680–689. Association for Computational Linguistics (2016). https://doi.org/10.18653/v1/D16-1065, http://www.aclweb.org/anthology/D16-1065

A study on improving accuracy to predict confidential words in judicial precedents using the neural network

Masakazu Kanazawa[1], Atsushi Ito[1], Kazuyuki Yamasawa[2], and Takehiko Kasahara[3]

[1] Utsunomiya University, 7-1-2 Yoto, Utsunomiya-shi, Tochigi, 105-0123 ,Japan
kanama2591@gmail.com, at.ito@is.utsunomiya-u.ac.jp
[2] TKC corporation, Karukozaka-MN Bldg. 2-1
Ageba-Cho,Shinjuku-ku,Tokyo,162-0824 ,Japan
yamasawa-kazuyuki@tkc.co.jp
[3] Toin Yokohama University, 1614 Kuroganecyo, Aoba-ku, Yokohama-shi, Kanagawa, 225-8503 ,Japan
kasahara@toin.ac.jp

Abstract. ICT (information and communications technology) has been adopted in the judicial fields of multiple countries. In Japan, the courts do not open all judicial precedents to the public due to privacy issues. In available precedents, confidential words, e.g., personal, corporate, and place names, are manually replaced by a single letter. This replacement process takes time and effort; therefore, not all precedents are opened. To solve this problem, we introduce a technique to automatically predict confidential words using a neural network. In a previous study, we demonstrated that our method is effective for predicting confidential words. However, the ac-curacy of the prediction was not sufficient for practical use. Therefore, we attempted to improve the detection accuracy for confidential words. First, we updated the parameters of the proposed model, such as the window size. Specifically, we enlarged the window size; this led to a better accuracy when detecting target words; however, the prediction accuracy for the confidential words decreased. In addition, prior to the learning in the neural network, we modified the input dataset to easily predict the target words. Unfortunately, using this method, we could not obtain good accuracy. Therefore, it is likely that the task of predicting the confidential words might be difficult when only using a neural network. One possibility to improve the accuracy is to use a proper noun dictionary. We explain this idea in detail in this paper.

Keywords: cyber court · judicial precedent · confidential word · neural network · proper noun dictionary

1 Introduction

Because the business world is increasingly recognizing the Internet as a place where commercial dealings can take place, the legal system needs to be equipped

to not only understand the Internet but also to use it as a tool to successfully exchange and use information. The Internet now provides a wide range of legal information, and one of the benefits of information being provided in this way is that it can be kept up-to-date as the law changes. Not only can the Internet assist in legal research but it can also assist in court processes in general, that is, in trial preparation and in the courtroom throughout the hearing. Reasons that courts should embrace such technology lie in the constantly increasing caseloads, the complexity of cases and jurisdictions, resource constraints, the pressure to improve access to justice, expectations of improvements in performance, and the pressure to improve efficiency and effectiveness in the court's administration and the delivery of justice [1]. Therefore, jurisdictions of many countries aim to follow economic growth and political changes to allow anyone to easily access judiciary documents by introducing ICT (information and communications technology). Such a justice system empowered by ICT is called a "cyber court". The pioneer study of a cyber court system is Courtroom 21 [2]; which started in 1993 at the College of William & Mary as a joint project between the university and the National Center for State Courts in the U.S.A. They implemented a cyber court using a teleconference system and performed multiple trials to determine the effectiveness of ICT in the court. In addition, the Singaporean Supreme Court introduced ICT in 1998 via measures such as a centralized display management system, a digital transcription system, e-signatures, electronic hearings, and an online justice system. All documents are displayed on PC screens in the court [3]. The High Court of Delhi and the District Court of Delhi have also introduced paperless e-courts. From the standpoint of technological research, there are many studies focusing on models combining law and the knowledge structures of law [4, 5]. Even China began stream some trials in more traditional courtrooms online in 2016 in an apparent effort to boost the transparency of their legal system [6]. Accordingly, we decided that it is very important to develop a prototype of a cyber court to evaluate the effectiveness of such an idea. In Japan, the prototype for the first civil trial was developed at the Toin University of Yokohama in 2004 [7, 8], and its effectiveness, particularly its usefulness in the Saibanin system (the Japanese jury system), was proven [9]. An experiment with a remote trial was also conducted [10]. The Investments for the Future Strategy 2017 by the Japanese Cabinet Office includes ICT conversion for trials to accelerate them and improving their efficiency [11]. From the viewpoint of using big data, this field is expected to develop further. The disclosure of judicial precedents is indispensable for using big data in AI and in the field of law; however, most precedents are not publicly open on Japanese court webpages. This is because Japanese privacy stipulates that an individual not be specified. Online privacy in Japan is primarily governed by a general law, the Act on the Protection of Personal Information (APPI), rather than a specialized law for online privacy. The APPI applies to all business operators that possess individuals' personal information. Japan has other personal information protection laws that apply to government and public organizations. Confidential words (e.g., personal, corporate, and place names) in opened precedents are replaced by other words, such as a single uppercase letter.

This operation takes time and effort because it is done manually. In addition, because trials involving people from various countries have been increasing due to globalization in recent years, creating a proper noun dictionary is difficult. However, neural networks have advanced greatly in the field of natural language processing in recent years. Studies including deriving a vector considering the meaning of words and predicting words are actively ongoing [12]. Therefore, we proposed two neural network models: a bi-directional LSTM (Long Short-term Memory) LR (left right) model and a Sum-LSTM based on the CBOW (continuous bag-of-words) model. We experimented with these models to ascertain their effectiveness for predicting confidential words. As a result, we were able to predict confidential words but did not obtain a good accuracy. In this paper, we review our experimental results and then we experiment with the proposed neural network by changing the window size and choosing proper parameter values. Then, we enhance the preprocessing of the input datasets. This operation is important for the neural network learning process. We evaluate the results of the experiment and propose a new model combined with a proper noun dictionary to form a more powerful method to improve the accuracy. In Section 2, we explain our initial experimental results. In Section 3, we discuss the result of an experiment with a modified window size. In Section 4, we discuss using a proper noun dictionary to increase the accuracy for predicting confidential words. In Section 5, we summarize our study and discuss future avenues of research.

2 Review of our comparative experiment on predicting the confidential words in Japanese precedents

2.1 A comparative experiment of four prediction method and the experiment

Japanese precedents include many confidential words such as personal names, corporate names, and place names. Due to the Japanese understanding of privacy, such words are converted into other words to protect privacy. In Japan, this procedure takes time and effort because it is done manually. Some judicial precedents are available on the website of the court [13]. Confidential words in these precedents have been replaced with an uppercase letter of the alphabet. (In paid magazines and websites, Japanese letters are sometimes used.) Figure 1 shows an example of a replaced word. This process is performed manually. To do these process automatically this processes, we proposed the following neural network models shown in Fig. 2. We experimented with the two proposed models to determine the prediction effect for confidential words. The two models are the bi-directional LSTM-LR model, which has been shown to be effective for modeling and predicting sequential data (see Fig. 2(b)) and the Sum-LSTM model, which is based on CBOW, which can represent the similarity of a word (see Fig. 2(c)). In this experiment, we also tested a simple LSTM model (see Fig. 2(a)) and a Sum bi-directional LSTM-LR, which combined the Sum-LSTM and LSTM-LR models (see Fig. 2(d)), to compare to the two proposed models.

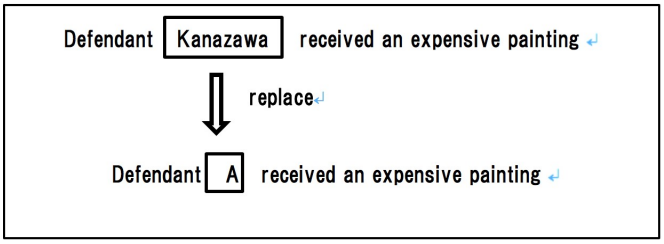


Fig. 1. Example of a replaced word

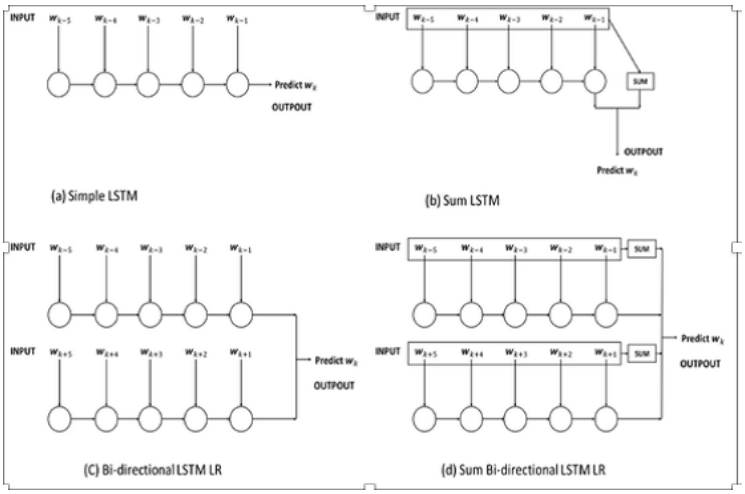


Fig. 2. The four neural network models used in the first experiment

2.2 Analyzing the result of the experiment

This experiment was conducted using the four models shown in Figure 2. Also, the embedding vector was 200 for all models. The judicial precedents used in this experiment are 20,000 precedents on the Japanese court website. Various parameters are shown in Table 1. For accuracy, we used perplexity (PPL) that was used in a previous research for predicting the next word. PPL was given by the following equation:

$$PPL = \frac{1}{P(correctword)} \tag{1}$$

In Eq. (1), P indicates the probability and PPL represents the number of prediction choices that are narrowed down to neural networks. The smaller the value, the better the prediction results. CW_PPL is the average PPL of the test data and reflects the perplexity of the confidential words, while PPL is the average for all the test data. The PPL scores are shown in Table 2. The result indicates that the two proposed models were both effective with respect

Table 1. Parameters of the model

Embedding size	200
Window size	10
Batch size	200

Table 2. The result of the first experiment

	Simple LSTM	Bidirectional LSTM-LR	Sum-LSTM	Sum-Bidirectional LSTM-LR
PPL	4.851	4.656	4.603	**4.462**
CW_PPL	56.277	**37.343**	72.463	77.031

Proposed model

to the PPL in our corpus. However, when we look at the results of CW_PPL, which is directly related to the task of predicting the confidential words, we find that the difference in the scores was at least 32.687 between PPL and CW_PPL in the case of the bi-directional LSTM-LR model. This result suggests that the task of predicting confidential words is more difficult. The PPL scores indicate that all the proposed methods are more effective than the simple LSTM model. However, for the prediction of confidential words, only the bi-directional LSTM-LR model showed good results. Sum-LSTM based on CBOW might have produced these results. CBOW is an effective model for paraphrasing words, and Sum-LSTM uses this mechanism. Therefore, when Sum-LSTM predicted a word that was "confidential", the CW_PPL score became worse due to the possibility of paraphrasing words such as "plaintiff", "defendant", "doctor", and "teacher". Knowing the paraphrased versions of the confidential words meant that embedding vectors of the confidential words could be successfully generated and that the model could recognize the meaning of "confidential". However, the prediction accuracy did not improve; therefore, it is possible that there might be problems when calculating the probability of the prediction task. In summary, the first experiment showed that our proposed models using neural networks were effective for predicting confidential words; nevertheless, the accuracy of these models need to be improved [14].

3 Changing the window size

3.1 Problems with the first experiment

We had the following problems in the first experiment for the bi-directional LSTM-LR model, which scored the best score compared to the other models. (1) The PPL score was poor for continuous numeric words. Continuous numeric

words scored approximately ten times worse than the target words. It is likely that our neural network model had a weakness for the continuous numeric words to train the many test dataset involved numeric words.
(2) The CW_PPL score was poor due to the possibility of paraphrasing words such as “plaintiff”, “defendant”, “doctor”, and “teacher”. To solve this problem, these para-phrasing words could be excluded from the choices when calculating the probability. In addition, it is important to examine scores other than PPL. These models were based on a paper by Mikolov [15]. However, there are other important parameters.

3.2 The aim of the experiment and the plan to improve the accuracy

Our ultimate goal is to obtain high accuracy when automatically predicting confidential words. Therefore, we solved the problem described in Section 3.1 and upgraded the detection accuracy for confidential words using a neural network. We designed an experiment based on the first experiment, that is, the bi-directional LSTM-LR model (Fig. 3). First, we review the parameters, such as the window size, of our neural network model to obtain the proper value for the window size. In addition, prior to the learning step in our proposed neural network, to predict the target words more easily, we enhance the input dataset via preprocessing by changing to a powerful morphological analyzer. Simultaneously, we avoid unnecessary words, i.e., “stop words”, in the input dataset. Finally, in the next section, we propose a method to improve the accuracy for predicting confidential words. In this method, we combine a proper noun dictionary with our proposed neural network.

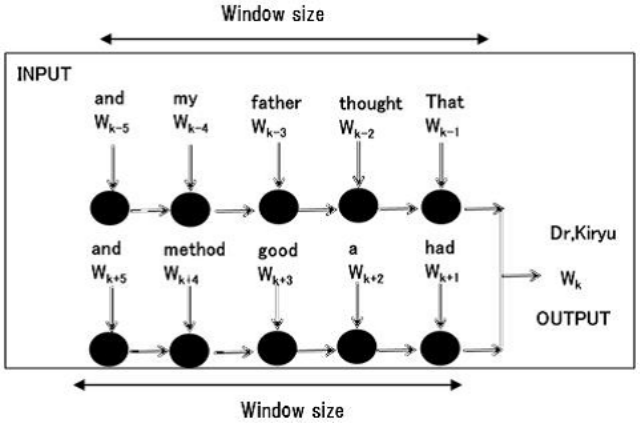


Fig. 3. the experiment using the model (Bi-directional LSTM LR)

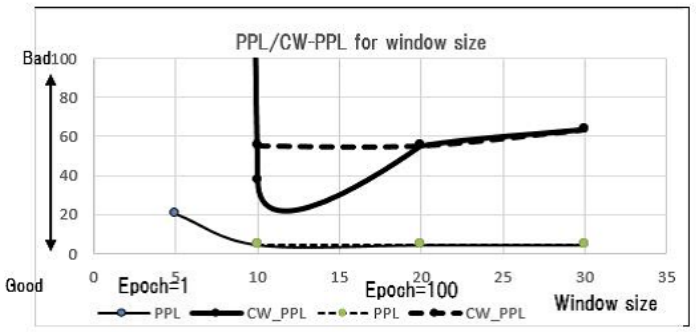


Fig. 4. Values of PPL and CW_PPL for different window sizes

3.2.1 Changing the window size

First, we focused on the window size. In general, a neural network such as LSTM has a chunk size (which we call the window size in this paper). The chunk size is the number of (output) frames for each chunk of data evaluated in the training or decoding. RNN or LSTM models, or "chain" models, are always trained on fairly large chunks (generally in the range of 40 − 150 frames). During decoding, we also generally evaluate the neural network using fairly large chunks of data (e.g., 30, 50, or 100 frames); this is usually referred to as the frames-per-chunk. For recurrent networks, the chunk-size/frames-per-chunk and the extra-left-context and extra-right-context are approximately the same during training and decoding because this generally gives the best results (even though sometimes it is the best to make the extra-context values slightly larger in decoding). One might expect that in decoding time longer context would always be better; however, this does not always seem to be the case [16]. If the length of a chunk is too short, the association between chunks cannot be learned, which will affect the accuracy of the LSTM model [17]. In our first experiment, we used a window size at 10 frames (words), as shown in Table 1. At that time, we did know if that this number was appropriate. Accordingly, we changed the "window size" parameter from 10 to 20 because we expected the PPL score to be improved by larger training samples. We expected to obtain the correct results [18]. In addition, we changed the window size from 10 to 5 to investigate how the training ability is affected by narrowing the window size. We see that the PPL values significantly worsen in this case. The PPL scores are seven times larger than those in the original experiment, and the CW_PPL values further worsened. Therefore, the window size greatly influences the accuracy when detecting confidential words. Next, we enlarged the window size from 10 to 20 or 30 and trained the neural network. The resulting PPL and CW_PPL scores are shown in Fig. 4. The PPL and CW_PPL scores are best when the window size is 20. When we enlarge the window size to 30, the PPL score is good but the CW_PPL score is worse. The CW_PPL score is likely poor when the window size is too large because, in this case, a large number of "confidential words" are contained in the window and

the neural network has many choices. We also changed the value of “epoch”. When training a neural network, one epoch indicates one pass through the full training set. Usually, a few iterations are used. We experimented with epoch values of 1, 5, and 100. Normally, when the epoch value is larger, the predicting accuracy is better. However, if the epoch value is too large, over-training will occur and the accuracy will decrease. In our experiment, even when the epoch value was 1, PPL changed little while CW_PPL worsened. We assume that an epoch value of 100 is better for predicting confidential words. The result is shown in Fig. 4.

3.2.2 Enhancement of the input dataset processing prior to learning

(1)Update of the morphological analyzer ”MeCab” to ”MeCab-ipadic-NEologd”

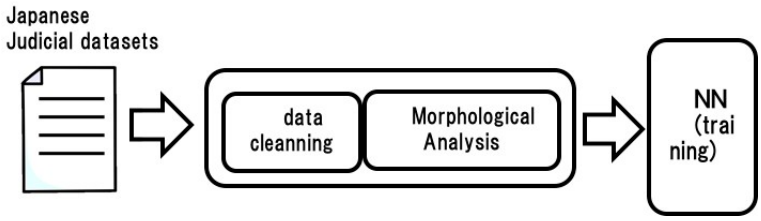


Fig. 5. Data preprocessing step

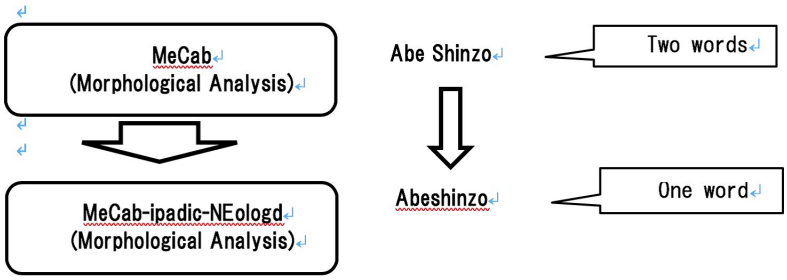


Fig. 6. MeCab upgrade

In the first experiment, we used 50,000 judicial precedents for the training data and 10,000 judicial precedents for the test data. These data included the records of trials from 1993 to 2017. We used the precedent database provided by TKC, a Japanese corporation [19]. Before these datasets flow into our proposed neural network, we converted all the confidential words contained in the datasets

to the uppercase letter “A” and separated the Japanese words with spaces using “MeCab”, a Japanese morphological analyzer [20], because we were using the Japanese judicial precedent dataset. These processes were performed in the data-preprocessing step (Fig. 5). In the second experiment, we decided to use “MeCab-ipadic-NEologd”, which is a more powerful morphological analyzer, has better performance than “MeCab”, and is an open software. We expected the prediction accuracy to improve for the confidential words (Fig. 6). However, the result of the experiment shows that the accuracy decreased compared to the first experiment (Table 3).

That is, there was no change in the PPL value but the CW_PPL value became

Table 3. **Result in the case of using mecab-ipadic- NEologd**

	Bidirectional LSTM-LR (mecab-ipadict-NEologd)	Bidirectional LSTM-LR (Mecab)
PPL	**4.78**	4.656
CW_PPL	**85.56**	37.343

two times worse. It is possible that, because the ability of the morphological analysis increased, overlearning occurred. In addition, the neural network parameters changed. Therefore, we need to continue the experiment to obtain the proper parameters, e.g., the epoch size and the window size.

(2) Legal terminology stop words

Because we are dealing with judicial words in the corpus, legal terminologies frequently appear in the corpus. These words do not necessary predict confidential words. Therefore, it is important to exclude these legal terminologies from the corpus. Accordingly, we registered these judicial words in a stop words list. Then, we removed these words from the corpus. The evaluation of the results will require further study.

4 Using a dictionary

4.1 Our new proposals to improve accuracy using the proper noun dictionary with our neural network model

The first experiment demonstrated that the task of predicting confidential words with high accuracy is difficult. We decided to improve the accuracy via a new model using a proper noun dictionary as input to the neural network. If the target word appears in the dictionary, it is easy to provide the correct answer. Even if the target word does not appear in the dictionary, if it is a confidential word, the input vectors will have a natural pattern. Therefore, we expect to

achieve high accuracy if we combine a dictionary with the prediction method. We display the flow of the current process, and that for future development, in Fig. 7.

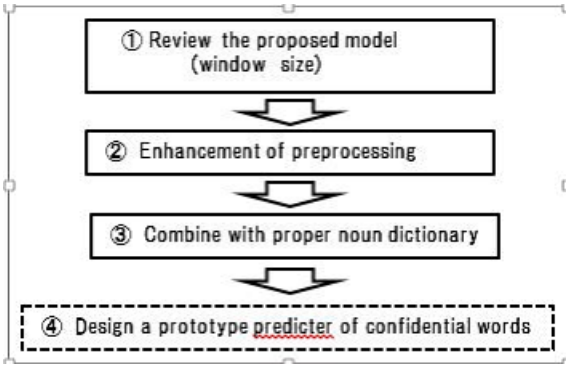


Fig. 7. Outline of this paper

4.2 Existing method of extracting the proper noun

Conventional proper noun extraction is performed using a named entity dictionary. The named entity extraction API extracts a named entity such as a person's name, a place name, or a date representation from the Japanese character string sent in the request. Then, we manually search for confidential words. These named entity words are useful for detecting confidential words. We provide an overview of this method in Fig. 8. This method eventually obtains the confidential words manually.

4.3 Our proposed method combined with a proper noun dictionary

4.3.1 One method using a proper noun dictionary

Confidential words in Japanese judicial words are usually a pronoun, such as personal name or a place name. We focus on this point. Accordingly, we use a proper noun dictionary with our neural network. There are several ways of using the dictionary. One method is to match the input corpus and the dictionary and then input the resulting data into the neural network (Fig. 9). Because this method is processed step by step, it is time intensive.

4.3.2 Using a proper noun dictionary

Conversely, our proposed method is processed directly because we combine the dictionary with the neural network. We describe the algorithm of our proposed model as follows. We use a proper noun dictionary, that is, the well-known morphological analyzer "MeCab-ipadic-Neologd" (MeCab-N), combined with our proposed neural network. When the corpus is input into MeCab-N, MeCab-N

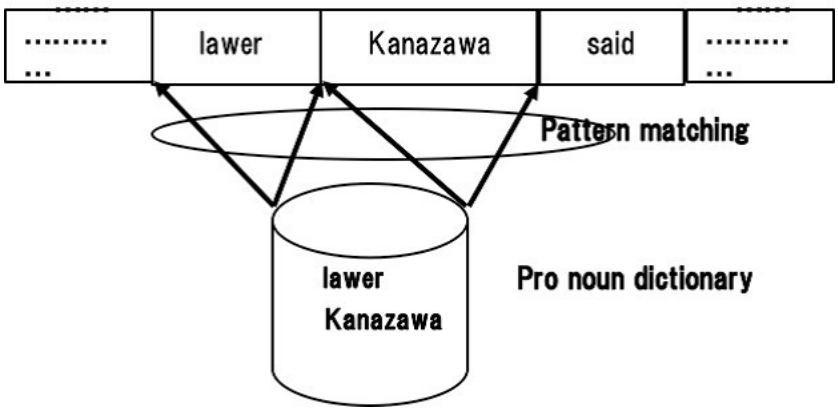


Fig. 8. Existing method of extracting the proper noun

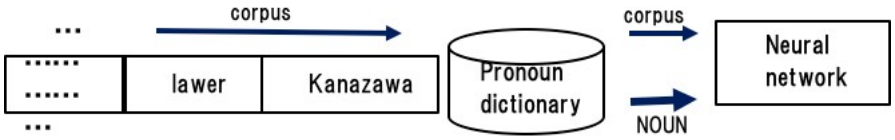


Fig. 9. using proper noun dictionary

disassembles the information into parts that contain a noun, verb, and adjective. Then, we choose the noun, e.g., the place name or the personal name, from the output data disassembled by MeCab-N. Next, we assign a unique index to the part information and make a part dictionary, merging them into the input data (word) for our neural network model via an embedding vector (Fig. 10).By using the proper noun dictionary information as input to the neural network, it is expected that the accuracy will increase. If the target word is in the proper noun dictionary, it is easy to identify. Even if the target word is not in the proper noun dictionary, if it is a confidential word, the input vectors are the same as the easy to recognize pattern. Therefore, we expect to obtain high accuracy. However, parameter adjustments for our proposed neural network are important because there have been no prior studies on such a model.

5 Conclusions

Privacy is one of the most important issues in cyber courts. However, it is not easy to completely hide confidential words such as personal information and locations. One technique to protect privacy is to find confidential words in a file or on a website and convert them into meaningless words. It would be useful to have a neural network that is intelligent enough to automatically anonymize

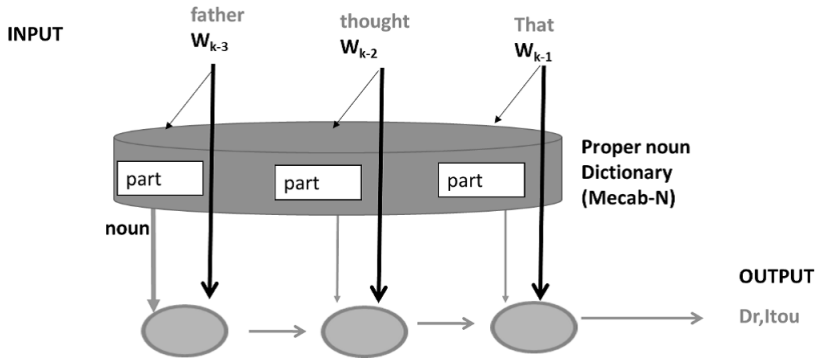


Fig. 10. Marging the proper noun dictionary with the neural network

confidential words. Based on Japanese judicial precedents, we have already proposed a recognition technique for confidential words using a neural network. In this paper, we proposed measures to improve the model' s prediction accuracy for confidential words. The first measure was to adjust the parameters of the proposed model to increase the accuracy. When we changed the window size of our proposed neural network to be larger and when we adjusted the epoch value, the PPL score was acceptable; however, the CW_PPL score was poor. This is because, in our method of replacing the confidential word, it is difficult to obtain the confidential word; therefore, we need to review and change our prediction method. The second measure was to improve the preprocessing. Prior to learning the input corpus, we enhanced the preprocessing of the corpus using the advanced analyzer "MeCab-ipadic-NEologd"; we hoped that this would improve the CW_PPL score; however, this was not the case for both PPL and CW_PPL. It is likely that a parameter adjustment is necessary. Finally, we proposed a new model that combines a proper noun dictionary with the neural network. This model has the ability to predict target words using our proposed neural network as well as the ability to match target words, such as confidential words, using a proper noun dictionary. If this model works well, we will establish an automatic detector tool for confidential words.

Acknowledgments We express special thanks to all people who joined the Cyber Court Project.

References

1. www7.scu.edu.au/1878/com1878.htm
2. http://law.wm.edu/academics/intellectuallife/researchcenters/clct/ [Sep, 5, 2017]
3. https://www.supremecourt.gov.sg/default.aspx?pgID=361No,10 [Sep, 5, 2017]
4. JURISIN, Japan, Oct 2013
5. JURISIN, Japan, Oct 2014

6. https://www.theverge.com/tech/2017/8/18/16167836/china-cyber-court-hangzhou-internet-disputes
7. Takehiko Kasahara: Cybercourt - Court Technology: Journal of the Japanese Society for Artificial Intelligence, vol. 23, No. 4, pp 513-, Jul. 2008]
8. Takehiko Kasahara: Court Technology for Civil Procedure: Festschrift for the seventieth birthday of Prof. Takeshi Kojima Ⅱ, pp. 961-, Sep. 2009
9. Can you judge?: NHK Special, 2005.2.13, PM9:00- : http://www6.nhk.or.jp/special/detail/index.html?aid=20050213 [Sep, 5, 2017]
10. A study of applying ICT for legal service: Report of a research funded by the Minis-try of Ministry of Internal Affairs and Communications in Japan in 2009, (March 2010)
11. http://www5.cao.go.jp/keizai-shimon/kaigi/minutes/2017/0609/shiryo_07.pdf
12. LAI, Siwei, et al. How to generate a good word embedding. IEEE Intelligent Systems, 2016, 31.6: 5-14 http://www.courts.go.jp/app/hanrei_jp/search1 [Sep, 5, 2017]
13. http://www.courts.go.jp/app/hanrei_jp/search1 [Sep, 5, 2017] Yuya Kiryu, Atsushi Ito, and Masakazu Kanazawa: Acta Polytechnica Hungarica [Sep9 2018]
14. Yuya Kiryu, Atsushi Ito, and Masakazu Kanazawa: Acta Polytechnica Hungarica [Sep9 2018]
15. MIKOLOV, Tomas, et al. : Efficient estimation of word representations in vector space: arXiv preprint arXiv:1301.3781, 2013. http://www.tkc.jp/law/lawlibrary [Sep 6, 2017]
16. http://kaldi-asr.org/doc/dnn3_scripts_context.html
17. LSTM-Based Hierarchical Denoising Network for Android Malware Detection (Security and Communication Networks Vol. 2018, Article ID 5249190, 18 pages)
18. Masakazu KANAZAWA, Atsushi ITO, Yuya KIRYU, et al.: Improve accuracy of predicting confidential words in judicial precedents: JURISIN, Japan, Oct 2014 Technical Committee on Thought and Language (TL) [July,28,2018] LSTM-Based Hierarchical Denoising Network for Android Malware Detection (Security and Communication Networks Vol. 2018, Article ID 5249190, 18 pages)
19. http://www.tkc.jp/law/lawlibrary [Sep 6, 2017]
20. http://taku910.github.io/mecab/ [Mar 9, 2018]

Using clustering techniques to identify arguments in legal documents

Prakash Poudyal, Teresa Gonçalves, and Paulo Quaresma

Department of Informatics, University of Evora
Evora, Portugal
prakashpoudyal@gmail.com, tcg@uevora.pt, pq@uevora.pt

Abstract. A proposal to automatically identify arguments in legal documents is presented. In this approach, cluster algorithms are applied to argumentative sentences in order to identify arguments. One potential problem with this process is that an argumentative sentence belonging to one specific argument can also simultaneously be part of another, distinct argument. To address this issue, a Fuzzy c-means (FCM) clustering algorithm was used and the proposed approach was evaluated with a set of case-law decisions from the European Court of Human Rights (ECHR). An extensive evaluation of the most relevant and discriminant features to this task was performed and the obtained results are presented.
In the context of this work two additional algorithms were developed: 1) the "Distribution of Sentence to the Cluster Algorithm" (DSCA) was developed to transfer fuzzy membership values (between 0 and 1) generated by the FCM to a set of clusters; 2) the "Appropriate Cluster Identification Algorithm" (ACIA) to evaluate the proposed clusters against the gold-standard clusters defined by human experts.
The overall results are quite promising and may be the basis for further research work and extensions.

Keywords: Fuzzy clustering algorithm, argument mining

1 Introduction

Advances in communication technology, accessibility of media devices and mushrooming of social media has caused the number of individuals expressing opinions to grow exponentially. As a result, massive amount of electronic documents are generated daily, including news editorials, discussion forums and judicial decisions containing legal arguments. In turn, rapid development of current research into argument mining is raising new challenges for natural language processing in various fields.

In general to automatically identify a legal argument within an unstructured text, three stages or modules are used in current practice. The first stage is to identify the argumentative and non-argumentative sentences, the second stage is to identify the boundaries of arguments and the third stage is to distinguish the argument's components (premise and conclusion).

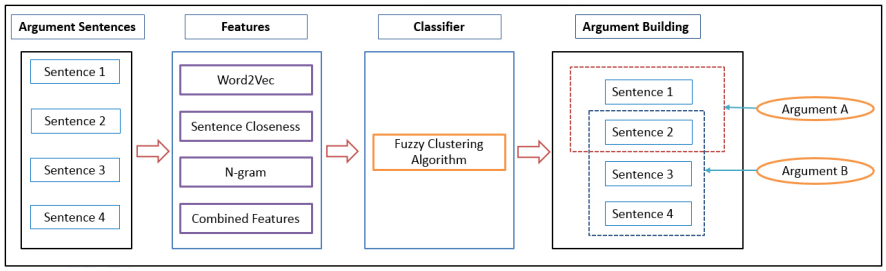


Fig. 1. Overview of the Architecture

To date, second stage processing has been performed by identifying the boundaries of arguments, an extensively explored method in the AI & Law literature. In this paper, we propose a clustering technique that groups argumentative sentences into a cluster of potential arguments with associated probabilities. An overview of our approach to the task is shown in Figure 1. The task is complex, since components of one argument (premise or conclusion) can also be involved in other arguments.

In the example shown in figure 1 there are 2 distinct arguments. In the example sentence 2 belongs to argument A and also to Argument B. To cluster such sentences, we propose to use a Fuzzy c-means (FCM) clustering algorithm [3] that provides a membership value ranging from 0 to 1 for each cluster of the sentence. These membership values are key assets of the FCM, since they allow us to associate each sentence with more than one cluster/argument. The performance of the FCM depends heavily on the selection of features that are used. In the context of our work we focused mainly on four kinds of features: N-gram, word2vec, sentence closeness and 'combined features'. Our aim is to identify the best performing set of features and techniques to cluster components to form an argument.

After extracting the features associated with each text, the FCM is used to obtain a cluster membership value for every sentence. To determine the composition of each cluster, we developed a specific algorithm: the "Distribution of Sentence to the Cluster Algorithm" (DSCA).

To evaluate the performance of the system, a second algorithm, the "Appropriate Cluster Identification Algorithm" (ACIA) was also developed to map each cluster of the system's output to the closest matching cluster in the gold-standard dataset.

The rest of the paper is organized as follows: section 3 contains a brief introduction to the datasets. Section 4 deals with measures used for evaluating the performance of the system. In Section 5 we describe the proposed architecture including a description of features, a discussion on determining the optimum number of clusters, and the newly developed DSCA and ACIA algorithms. Section 6 evaluates the performance of all of our experiments. Lastly, Section 7 addresses conclusions and prospects for future work.

2 State of the Art

In the argument mining field, there has been very limited research about using clustering techniques to identify and group argumentative sentences into arguments. However, similar problems have been solved by other researchers through the use of boundary detection techniques e.g. [14] looked at the places that signal where an argument begins and ends. Since components of an argument can be spread all over a text, these authors proposed the use of semantic distance to handle this problem via context-free grammars (CFG). This approach was applied to a strictly limited subset of case-law obtaining 60% accuracy. [22] proposed a novel annotation scheme to model arguments. Their approach was to identify the relation (i.e. 'support' or 'attack') between the components of arguments. Their technique indicates which premises belong to a claim and constitute the structure of arguments. [11] performed a manual analysis as well as an automated analysis to detect the boundaries of an argument. To train and test for automatic analysis, the authors relied on help from experts to analyze the text manually. For the automatic analysis, they used two Naive Bayes classifiers; one to identify the first word of a proposition and the other to identify the last word. [9], [20], and [17] continued this boundary approach, using the Conditional Random Fields (CRF) algorithm to segment the argument's components.

[23] is a survey paper about the use of machine learning technologies for argument mining. The paper analyses several approaches made by different authors regarding argument boundary detection. This review article emphasizes that the boundary detection problems depend upon the adopted argument models. The main reason is the existence of different types of argument component structure.

Conrad [7] applied a k-means clustering algorithm over plaintiff claims in 'Premises Liability', 'Medical Malpractice' and 'Racial Discrimination' suits. The authors applied their technique to distinguish more effective plaintiff claims from less effective ones using an 'award_quotient' metric to segregate the claims. Besides award_quotient, the authors used features to help differentiate one cluster's properties from another. The authors mention that they also tried aggregative, partial and graphical features, but didn't find anything that yielded a performance superior to k-means.

3 The Data Set

We selected case-law documents from the European Court of Human Rights (ECHR)[1] annotated by R. Mochales [15]. The corpus is composed of 20 Decision and 22 Judgment categories released before 20 October 1999 by the European Commission on Human Rights. Both categories include similar information, however, the 'Decision' present the information briefly with an average word length of 3,500 words, whereas for 'Judgment' the average word length is 10,000 words. We have 9257 sentences, out of which 7097 (77%) of them are

[1] http://hudoc.echr.coe.int/sites/eng

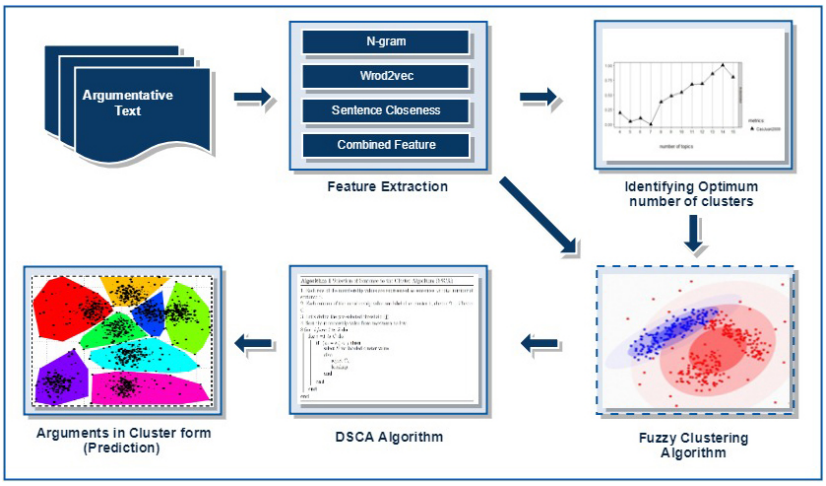


Fig. 2. Proposed Architecture of the Clustering System.

non-argumentative and 2160 (23%) argumentative sentences. Details about the ECHR corpus is available in [15].

4 Evaluation Setup

We used Precision, Recall, F-measure [2, 19] in the analysis of our clustering technique. Furthermore, we also used cluster purity to evaluate the quality of the obtained clusters. We computed the cluster purity [21] by counting the number of correctly assigned entities and dividing by the total number of N. Formally

$$ClusterPurity\,(\varphi, C) = \frac{1}{N}\sum\nolimits_{d=1..k} max_{e=1..k}|w_d \cap c_e| \qquad (1)$$

where N is the summation of the total number of elements in all clusters, $\varphi = \{w_1, w_1, \cdots w_k\}$ is the set of clusters and $c = \{c_1, c_1, \cdots c_k\}$ is the set of classes. We interpret w_d as the set of sentences in w_d and c_e as the set of sentences in c_e in Equation 1.

5 System Architecture

Our proposal is to cluster argumentative sentences and, thereby, to identify legal arguments. Figure 2 presents the system architecture. There are several phases: first: feature extraction; second: identifying the optimum number of clusters in the respective case-law file; third: converting the soft clustering values generated by FCM to hard clustering; and fourth (and final stage): system evaluation.

5.1 Feature Extraction

Typically features are values that represent a sentence and are suitable for a machine learning algorithm to handle. It is essential to select the most appropriate and precise features to train a machine learning algorithm so that the model can be successfully applied to new data. Therefore, good discriminant features are needed to correlate similarities between sentences and also address the sequential nature of sentences (since the majority of the components of an argument are presented in order). To address this requirement, the following features were used: N-gram [5], word2vec, and sentence closeness (discussed below). Another feature set can also be obtained by combining these three existing features into what we called "Combined Features". Each kind of features is discussed below.

Word2vec: The word2vec approach was proposed by [13] and can be implemented in two different ways: as a 'Continuous Bag of Words' (CBW) or as a 'Skip gram'. With Skip-grams, context words are predicted from selected words in the text, whereas with CBW, a word vector is predicted from the context of adjacent words. A Wikipedia dump of 05-02-2016 was used as input to the word2vec implementation of Gensim [18], where 100 dimension vectors were generated for each word. From the training set, each word of the sentence is looked up and its corresponding vector found among the generated word vectors. Then, the average of all vectors of the words presented in the sentence is taken and considered to be the 'sentence vector'.

Sentence Closeness: Sentence closeness is the reciprocal of the inter-sentence distance (i.e. the distance between sentences) counted in units of whole sentences. To capture the sequential nature of sentences, distance is a useful feature that helps to determine which sentences belong to which argument. The highest scoring sentence is considered to be the origin sentence (with a score of 1) from which all other distances are measured. With the exception of the origin sentence, 'closeness' scores should decrease monotonically as they move away from the origin. Furthermore, as meaning and concepts flow from one sentence to another, this implies that sentences whose 'closeness' is high are good candidates for being clustered together i.e. they belong to the same argument. Equation 2 was used to calculate the 'closeness' for each sentence.

$$Closeness(n) = \frac{1}{n} \tag{2}$$

where n the distance between sentences, measured in integer units of sentences.

Combined Features: The previously presented features (N-gram+'Sentence Closeness'+Word2vec) were combined into a new feature in an attempt to improve the performance of the clustering algorithms.

5.2 Identification of the optimum number of clusters

An argument cluster is a set of sentences which together comprise a single, coherent legal argument. The process by which sentences are aggregated into arguments in this way is called clustering. To cluster sentences successfully into

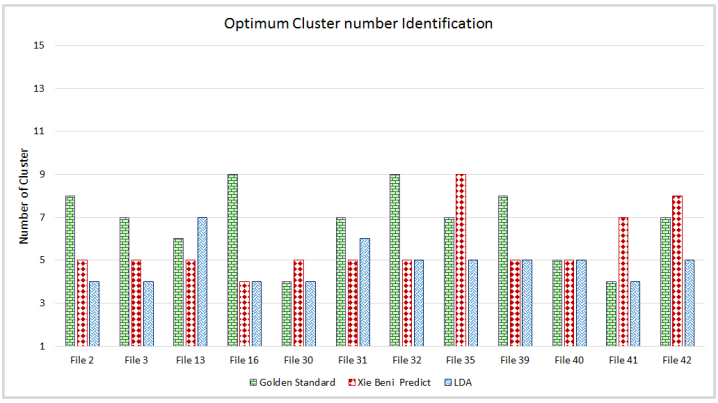


Fig. 3. Argument numbers of gold-standard vs. System Prediction (proposed by Xie and Beni and Cao *et al.*)

arguments, it is currently necessary to specify in advance how many clusters to expect within a corpus and until recently, there has been no well-established approach to defining this. Techniques that claim to be able to define the optimum number of clusters in the FCM have been proposed by [24] and Latent Dirichlet Allocation (LDA) [4].

Employing the Xie and Beni approach [24], we determined experimentally that an FCM with a fuzziness value of m set to 1.3 in concert with the features Word2Vec, N-gram and Sentence Closeness, yielded the best results with our particular set of case-law files. [24] technique selects the best candidate for the number of clusters after obtaining the minimum index value from that respective cluster number.

The Latent Dirichlet allocation (LDA) technique estimates the number of 'topics' existing within a text, which means estimating the probabilities of groupings within the text. Inspired by the concept, it was decided to look for such groups within our own corpora, and to use the estimated number of topics as a proxy for the number of clusters. Unfortunately, the LDA technique does not accept the Word2Vec, TFIDF, and Sentence Closeness features, and so, 'word frequency' in the document was used instead. We selected the 'CaoJuan2009' method as a metric which is the best LDA model based on density [16]. 'CaoJuan2009' was tested and agreed well with the number of topics (equivalent to our 'number of clusters') predicted by the minimum index value [6].

Figure 3 illustrates the results for each experiment: the first is the gold-standard, the second is using Xie and Beni's proposal, and the third is from LDA [6]. In the case of Xie and Beni, it can be observed that case-law files 02, 31, 32, 39, 42 find the closest number of clusters to the gold-standard, whereas for other case-law files the differences are greater.

In case of Cao *et al.*'s prediction: Case-law files 40 and 41 finds the correct data required for identification, whereas other case-law files present a slight

difference, but not as big as that observed by Xie and Beni. The exact accuracy score achieved was an identical 8% for both LDA and Xie and Beni.

We also used equation 3, to calculate the difference between the number of clusters of the gold-standard (C_g) and the ones predicted by our system (C_s). If they differ only by one unit then we considered the prediction is "almost correct"; otherwise it's an incorrect prediction. The result of applying this filter shows that the accuracy of the closeness scores increase in value to 58% for LDA and 42% for Xie and Beni, respectively.

$$|C_s - C_g| \leq 2 \tag{3}$$

where C_s is the cluster number given by the prediction system, and C_g is the cluster number of the gold-standard.

From the analysis, it's possible to conclude that LDA achieves a greater accuracy than Xie and Beni and is much closer to the gold-standard. Improvements in the results are expected to be achieved in the future as more discriminant features are used.

5.3 Clustering Algorithm

After extracting the features, we used a standard Fuzzy c-means (FCM) Clustering algorithm [3] to generate membership values ranging from 0 to 1 for each cluster. The number of clusters was defined based on the algorithm proposed and described in section 5.2. We set the fuzziness value $m \in \{1.1, 1.3, 2.0\}$.

5.4 Distribution of Sentence to the Cluster Algorithm (DSCA)

The Distribution of Sentence to the Cluster Algorithm (DSCA) algorithm aims to transform the membership value generated by FCM (between 0 and 1) into the set of clusters, each with a membership probability indicating how likely it is that the sentence belongs to a particular cluster. DSCA is presented as Algorithm 1

Membership values are represented by a matrix where each row represents a sentence and each column is labelled with a cluster number (C_i) ranging from 1 to C. To assign a sentence to the respective cluster, a threshold value t needs to be specified to help define boundaries between the clusters. The value is defined only if the difference between the maximum membership value of the i^{th} position is less than the threshold value for the cluster, otherwise the sentence is rejected. The algorithm ends after conducting an iterative process through all positions in the matrix. The concept of *threshold value* is discussed by [1] as well as [10]. [1] used a *threshold value* to eliminate the data points. The authors also claim that the definition of the appropriate *threshold value* was determined by experimentation. After applying the DSCA algorithm, we were able to obtain a proposal for legal arguments: the identified clusters and their sentences.

Algorithm 1: Distribution of Sentence to the Cluster Algorithm (DSCA)

1. Denote the matrix of the sentences x cluster by $(a_{ij}) \in [0, 1]$, $i = 1, 2, 3 \cdots S$ and $j = 1, 2, 3 \cdots C$ such that i stands for sentence and j stands for cluster.
2. Pre-selected threshold (t) is defined
3. **for** *each* i *do* **do**
   $i_{max} = max(a_{ij}) \quad \forall i$
   for *each* j *do* **do**
      if $(i_{max} - a_{ij}) < t$ **then**
         select sentence i for cluster j
         else
            reject i;
         end
      end
   end

end

6 Result and Evaluation

In order to perform an evaluation of the performance of our system, we needed to find the best mapping between our system's clusters and the existing gold-standard data clusters from the ECHR. Therefore, we proposed and developed a new algorithm the "Appropriate Cluster Identification Algorithm "(ACIA) to solve this problem. This algorithm maps the argument predicted by our system to the closest matching argument in the gold-standard corpus. Here, we describe the details of the algorithm.

6.1 Appropriate Cluster Identification Algorithm (ACIA)

The ACIA algorithm aims to find the best mapping between the system's predicted clusters and the gold-standard dataset clusters. A formal description of the ACIA algorithm is presented in Appendix 8. Here, the steps of the algorithm are summarized.

Step 1: Initialize the cost value $C = 0$.

Step 2: Calculate the f_1 values between clusters in the gold-standard datasets and clusters in the system predicted datasets and arranged in matrix form.

Step 3: Select the maximum value from matrix. After value selection, the corresponding row and column containing the value are removed from future consideration.

Step 4: The selected value is considered as a node and is inserted with the cost value $C = 0$ into tree structure.

Step 5: Go back to Step 3 until there are no more eligible f_1 values in the matrix left to consider.

Step 6: The total cost of each route in the tree is then calculated. The route with the maximum cost is then selected.

Step 7: These routes consist of f_1 values for the best mapping between the system predicted clusters and the gold-standard sets.

After identifying the best mapping between the system clusters and the gold standard arguments, the f_1 measure is calculated for each cluster and the overall average f-measure value is obtained.

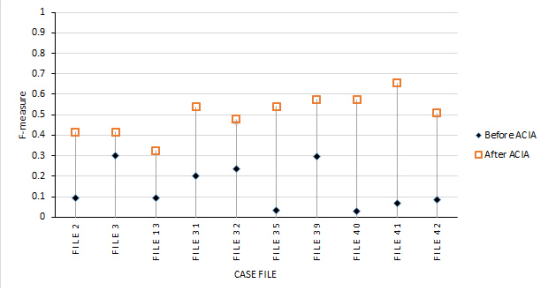


Fig. 4. F_1 score before and after applying ACIA

In figure 4 we can see the relevance of performing a optimized mapping between the system's predicted clusters and the gold-data arguments. We can observe that the value of 'After ACIA' (square symbol) is higher (above 0.3 for all files), whereas in the case of 'Before ACIA' (diamond symbol), the maximum value never exceeds is 0.3.

6.2 Performance Measurement

The experiment was conducted with the features mentioned in section 5.1 with the parameters of fuzziness parameter $m \in \{1.1, 1.3, 2.0\}$ and *threshold value* $t \in \{0.0001, 0.00001, 0.000001\}$ used for the conversion from a soft to a hard clustering. For the reason of space, we present the results for the features and parameters that score the highest f_1 value in most of the case-law files. Table 1 presents the performance result (precision, recall and f_1, cluster purity) showing the N-gram, Sentence closeness, Word2vec and 'Combine features' using a *threshold value* $t = 0.00001$ and FCM fuzziness $(m) = 1.3$. Along with this, we include the number of sentences of each case-law file. The highest f_1 value of each case-law file obtained from each feature is highlighted in bold and underlined. Case-law files 03, 13, 16, 31, 32 and 42 obtained the highest value using Word2vec features. Case-law file 02 scored the highest f_1 value with N-gram, and case-law files 30, 35 and 41, the highest f_1 with the combined features. Likewise, case-law file 40 scored the highest f_1 value with the Sentence Closeness feature. From this analysis, we can conclude that Word2vec seems to be the best overall approach.

In comparison to Word2vec, N-gram did not perform as well. The main reason for this effect is that N-gram uses a bag of words approach which is not effective in

Case	#S	N-gram				Sentence Closeness				Word2vec				Combined Feature			
		Pre	Rec	f_1	Purity	Pre	Rec	f_1	Purity	Pre	Rec	f_1	Purity	Pre	Rec	f_1	Purity
02	15	0.698	0.485	**0.573**	**0.625**	0.342	0.221	0.268	0.412	0.656	0.367	0.470	0.563	0.625	0.450	0.523	0.600
03	15	0.619	0.429	0.506	0.563	0.405	0.333	0.366	0.412	0.714	0.429	**0.536**	**0.600**	0.524	0.381	0.441	0.533
13	20	0.508	0.628	0.561	0.500	0.413	0.344	0.375	0.400	0.602	0.581	**0.591**	**0.550**	0.342	0.344	0.343	0.400
16	33	0.125	1.000	0.222	0.125	0.437	0.481	0.458	0.424	0.449	0.449	**0.449**	**0.424**	0.125	1.000	0.222	0.125
30	25	0.265	1.000	0.419	0.263	0.252	0.275	0.263	0.320	0.351	0.363	0.357	**0.360**	0.272	1.000	**0.428**	0.275
31	15	0.317	0.714	0.439	0.313	0.524	0.571	0.547	0.533	0.595	0.524	**0.557**	**0.533**	0.429	0.500	0.462	0.400
32	17	0.335	0.785	0.470	0.326	0.481	0.393	0.433	0.474	0.648	0.485	**0.555**	**0.529**	0.500	0.396	0.442	0.474
35	13	0.429	0.414	0.421	0.467	0.619	0.414	0.496	0.571	0.667	0.414	0.511	0.615	0.845	0.636	**0.726**	**0.769**
39	17	0.352	0.588	**0.440**	0.346	0.400	0.431	0.415	**0.421**	0.362	0.525	0.429	0.368	0.310	0.613	0.412	0.250
40	14	0.400	0.370	0.384	0.467	0.587	0.530	**0.557**	0.533	0.519	0.520	0.520	**0.533**	0.400	0.420	0.410	0.467
41	12	0.517	0.563	0.539	0.500	0.625	0.625	0.625	**0.583**	0.438	0.438	0.438	0.417	0.683	0.625	**0.653**	**0.583**
42	18	0.464	0.440	0.452	0.389	0.433	0.414	0.424	0.389	0.643	0.486	**0.553**	**0.500**	0.431	0.598	0.501	0.414

Table 1. Precision, Recall and f_1, Cluster Purity value according to Case-law and the number of sentences

finding similarities between sentences, and the results show that the performance of N-gram depends upon the number of sentences; if the number of the sentences in the case-law file is high, then N-gram performance is poor.

Sentence Closeness is another important feature that helps to understand the sequential context of the sentence. The sentence following an argumentative sentence often has a huge impact on the argument, as the meaning/context of a sentence usually flows sequentially. The results in this table show that the performance of Sentence Closeness is satisfactory, but still lacking in comparison to Word2vec. The Combined feature also has an impact, as it is a combination of Word2vec, N-gram and Sentence Closeness. The Combined feature offered the highest f_1 value for those case-law files for which Word2vec did not offer significant results, with the exception of case-law files 02, 39 and 40. Overall, 66% of the case-law files obtained the highest f_1 using Word2vec and Combined feature.

Furthermore, in the case of N-gram and the Combined feature, we found recall is further elevated by up to 1, but precision is very low for case-law files that have a large number of sentences. This is because the n-gram feature is inappropriate for such case-law files. If the feature is not sufficiently discriminating enough to distinguish among sentence categories, applying the FCM provides equal membership probability values (or very close to equal) for every category, essentially providing no useful information. As a result, during the process of forming hard clusters, such sentences get equally distributed over all clusters. During the evaluation process, this results in recall scores that are very high because a large number of sentences will be retrieved from many categories, but this abundance of hits in many categories also implies that the precision will be low.

Table 1 also presents the cluster purity value of each feature obtained for each case-law file. Word2vec was found to play the leading role in case-law files 03, 13, 16, 30, 31, 32, 40, and 42. Sentence Closeness scored highest in four case-law files: 16, 31, 39 and 41. However, case-law 16 and 31 tied with Word2vec.

Overall the purity values are satisfactory, except in case-law file 16 and 33 with the Combined feature and N-gram. Case-law 16 which had 33 sentences had the lowest value (0.125) from Combined and N-gram features. Similarly, case-law 30, which has 25 sentences, obtained 0.275. On the other hand, case-law 35, which had 13 sentences, scored 0.726 (the highest value) using the Combined feature. From this analysis, we can conclude that having a greater number of sentences also affects the clustering quality negatively and that Word2vec is the dominant feature for obtaining acceptable f_1 and cluster purity values. It is apparent from the data in Table 1 that f_1 and cluster purity are well correlated.

Overall, the results obtained from the proposed framework are promising even if they cannot be compared with other researchers' results. [14] had a similar objective to our own, and obtained 60% accuracy while focusing on argumentation structure detection. However, since different performance measurements are used, our results cannot be compared to theirs. Our approach, which can be used for any corpora, is not limited to a specific domain. Our results are much closer to [14] than [9], who obtained an accuracy of 42% while segmenting the argumentative sentence using Conditional Random Fields (CRF), or [11], whose precision and recall for identifying argument structure using automatically segmented propositions were 33.3% and 50.0%, respectively. Moreover, their manually segmented propositions reached precision rates of 33.3% and recall rates of 18.2%. [22] also encountered problems dealing with 'support' and 'attack' relations. The main reason for this was that their approach was unable to identify the correct target of a relation, especially in a paragraph with multiple claims or reasoning chains.

7 Conclusion and Future Work

We proposed a new clustering technique for grouping argumentative sentences in legal documents. We also proposed and implemented an evaluation procedure for the proposed system. Moreover, we also proposed an approach to identify the total number of arguments in a case-law document. Overall, the results that we achieved are satisfactory. The macro f_1 and average cluster purity score for system prediction using Word2vec feature in case-law files that have 4 to 8 arguments is 0.497 and 0.499 respectively.

For future work, we propose adding more features, such as 'semantic similarity' to try to improve these results. Moreover, as an extension of this work we are working on the identification, in each cluster/argument, of sentences acting either as a premises or conclusions.

References

1. Moh'd Belal Al-Zoubi, Amjad Hudaib, and Bashar Al-Shboul. A fast fuzzy clustering algorithm. In *Proceedings of the 6th WSEAS Int. Conf. on Artificial Intelligence, Knowledge Engineering and Data Bases*, volume 3, pages 28–32, 2007.

2. Ricardo Baeza-Yates and Berthier Ribeiro-Neto. *Modern information retrieval*, volume 463. ACM press New York, 1999.
3. James C. Bezdek, Robert Ehrlich, and William Full. Fcm: The fuzzy c-means clustering algorithm. *Computers & Geosciences*, 10(2):191 – 203, 1984.
4. David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.
5. Peter F Brown, Peter V Desouza, Robert L Mercer, Vincent J Della Pietra, and Jenifer C Lai. Class-based n-gram models of natural language. *Computational linguistics*, 18(4):467–479, 1992.
6. Juan Cao, Tian Xia, Jintao Li, Yongdong Zhang, and Sheng Tang. A density-based method for adaptive lda model selection. *Neurocomputing*, 72(7-9):1775–1781, 2009.
7. Jack Conrad and Khalid Al-Kofahi. Scenario analytics system, December 21 2017. US Patent App. 15/618,920.
8. Douglass R Cutting, David R Karger, Jan O Pedersen, and John W Tukey. Scatter/gather: A cluster-based approach to browsing large document collections. In *Proceedings of the 15th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 318–329. ACM, 1992.
9. Theodosis Goudas, Christos Louizos, Georgios Petasis, and Vangelis Karkaletsis. Argument extraction from news, blogs, and social media. In *Hellenic Conference on Artificial Intelligence*, pages 287–299. Springer, 2014.
10. Anil K Jain, M Narasimha Murty, and Patrick J Flynn. Data clustering: a review. *ACM computing surveys (CSUR)*, 31(3):264–323, 1999.
11. John Lawrence, Chris Reed, Colin Allen, Simon McAlister, and Andrew Ravenscroft. Mining arguments from 19th century philosophical texts using topic based modelling. In *Proceedings of the First Workshop on Argumentation Mining*, pages 79–87, 2014.
12. Yanjun Li, Soon M Chung, and John D Holt. Text document clustering based on frequent word meaning sequences. *Data & Knowledge Engineering*, 64(1):381–404, 2008.
13. Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.
14. Raquel Mochales and Marie-Francine Moens. Argumentation mining. *Artificial Intelligence and Law*, 19(1):1–22, 2011.
15. Raquel Mochales-Palau and Marie Francine Moens. Study on sentence relations in the automatic detection of argumentation in legal cases. *FRONTIERS IN ARTIFICIAL INTELLIGENCE AND APPLICATIONS*, 165:89, 2007.
16. Nidhi. Number of topics for lda on poems from elliston poetry archive. Available at `http://www.rpubs.com/MNidhi/NumberoftopicsLDA` (2017-03-31), 2017.
17. Joonsuk Park, Arzoo Katiyar, and Bishan Yang. Conditional random fields for identifying appropriate types of support for propositions in online user comments. In *Proceedings of the 2nd Workshop on Argumentation Mining*, pages 39–44, 2015.
18. Radim Řehůřek and Petr Sojka. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta, May 2010. ELRA. `http://is.muni.cz/publication/884893/en`.
19. M.E.S. Rodrigues and L Sacks. A scalable hierarchical fuzzy clustering algorithm for text mining. In *Proceedings of the 5th international conference on recent advances in soft computing*, pages 269–274, 2004.

20. Christos Sardianos, Ioannis Manousos Katakis, Georgios Petasis, and Vangelis Karkaletsis. Argument extraction from news. In *ArgMining@ HLT-NAACL*, pages 56–66, 2015.
21. Hinrich Schütze, Christopher D Manning, and Prabhakar Raghavan. *Introduction to information retrieval*, volume 39. Cambridge University Press, 2008.
22. Christian Stab and Iryna Gurevych. Annotating argument components and relations in persuasive essays. In *Proceedings of the 25th International Conference on Computational Linguistics (COLING 2014)*, pages 1501–1510. Dublin City University and Association for Computational Linguistics, August 2014.
23. Marco Lippi and Paolo Torroni. Argumentation mining: State of the art and emerging trends. *ACM Trans. Internet Technol.*, 16(2):10:1–10:25, March 2016.
24. Xuanli Lisa Xie and Gerardo Beni. A validity measure for fuzzy clustering. *IEEE Trans. Pattern Anal. Mach. Intell.*, 13(8):841–847, August 1991.

8 Appendix

8.1 Appropriate Cluster Identification Algorithm (ACIA)

Let A be the system's cluster set and the B the gold standard cluster set, respectively, having a cardinality of n: $A = \{a_1, \cdots, a_n\}$ and $B = \{b_1, \cdots, b_n\}$. We define the matrix $F = \{f_{ij}\}$ where $f_{ij} = a_i b_j$ with $a_i \in A$ and $b_j \in B$. Here, $F = \{f_{ij}\}$ is the f-measure value calculated by taking cluster i from A and cluster j from B.
We denote by $(F)_{ij}$ the matrix formed from the matrix F by removing the j^{th} column and i^{th} row

State 1 : Initialization https://v2.overleaf.com/project/5b7d5fba95426b6bb8c66ef6

$$F^o = (f_{ij})_{n \times n}$$

$$R^o(-1, -1) = \emptyset$$

i.e. Nodes are connected with the cost value C=0 to form a tree structure.

State 2 : From $k = 0$ to n, iterate. At each k step, we have $F^{(k)}(i, j)$ and $R^{(k)}(i, j)$

Find all maximum elements of $F^{(k)}(i, j)$
Let $M_k = \left\{(i, j) | f_{ij}^{(k)} is\ the maximum\ element\ of\ F^{(k)}(i, j)\right\}$
i.e. Maximum f-measure value is selected and place in tree structure;

State 3: For each element $(i, j) \in M_k$, update route

$$R^{(k+1)}(i, j) = R^{(k)}(i, j) \cup \{(i, j)\}$$

and matrix

$$F^{(k+1)}(i, j) = \left(F^{(k)}\right)_{ij}$$

Do it for all elements (i, j) of M_k
Stop when $k = n$.

i.e. Procedure repeat again for other remaining values;

State 4 : {For each route, calculate total cost}

$$TC_{R^{(k)}(i,j)} = \sum_{(i,j)\in R^{(k)}(i,j)} f_{ij}$$

i.e. The total cost of each route is calculated.

State 5: Select one of the maximum values

$$TC_{R_o(i,j)},$$

and its route

$$R_o(i, j) = \{(i_1, j_1), \cdots, (i_n, j_n)\}$$

i.e. The route with the maximum scores is selected.

After identifying the appropriate cluster (argument) with respect to the gold-standard; an f-measure is calculated between the i^{th} cluster of the system as recommended by the ACIA and the j^{th} cluster of the gold-standard. After that, the average f-measure value is calculated.

ContractFrames: Bridging the gap between Natural Language and Logics in Contract Law *

María Navas-Loro[1] (orcid.org/0000-0003-1011-5023), Ken Satoh[2], and Víctor Rodríguez-Doncel[1] (orcid.org/0000-0003-1076-2511)

[1] Ontology Engineering Group, Universidad Politécnica de Madrid, Spain
{mnavas,vrodriguez}@fi.upm.es
[2] Principles of Informatics Research Division, National Institute of Informatics
ksatoh@nii.ac.jp

Abstract. This paper introduces ContractFrames, a framework able to translate natural language texts referring to the different events related to the status of a purchase contract to logic clauses from a legal reasoning system called PROLEG. Diverse frames and rules have been developed for the extraction and storage of this information before its conversion to logic clauses. Our framework uses natural language tools and rules to extract relevant information, store it in the form of frames, and return the PROLEG version of the input text. Also an ontology, called the Contract Workflow Ontology, has been developed to represent all the relevant information of the events related to a contract. The framework has been tested in a syntethic dataset, and shows promising results.

Keywords: Legal NLP · PROLEG · Contract Life-cycle · Legal Ontology.

1 Introduction

Making machines to understand commercial contracts is a challenging and multidisciplinary task. Natural Language Processing (NLP) techniques are required to analyze different documents to extract relevant information, which can be expressed in very different formats and records. Once obtained, this information needs to be properly represented, via some knowledge representation system. Finally, reasoning methods are also required to extract new knowledge considering collected information.

Focusing on the legal domain, there exist different proposals for formalizing legal information in the form of logical predicates. We find among them PROLEG [25], a legal reasoning system able to represent and reason about contract status

* This work was partially supported by JSPS KAKENHI Grant Number 17H06103 and by a project with funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 780602. It has been also partially supported by a Predoctoral grant from the I+D+i program of the Universidad Politécnica de Madrid. This work has been done during an internship funded by the National Institute of Informatics.

and derive information such as its validity or the right or reason of a rescission, among others. Nevertheless, in spite of having all the contract law logic needed already coded, how to automatically transform the input text into logic facts remains still as an open and important challenge. Currently, the translation of texts describing contract events must be manually coded, being therefore very inefficient in terms of cost and time, and remaining unsolved the fact that any ulterior change would need manual curation. A bridge between NLP and this logical system is therefore needed to automatically retrieve all relevant facts from text to populate the PROLEG fact knowledge base.

In this paper, we propose a framework, called ContractFrames, able to translate natural language texts referring to the different status of a purchase contract into PROLEG clauses. These texts are not normative texts nor regular texts (being both types extensively studied in previous literature), but some natural language text at a mid point between regular language and pure legal language; an example of one of these texts can be found in Fig. 1, along with its translation into PROLEG. To the aim of expressing these texts in a full logical legal language, we have developed different frames[3] and rules for representing and extracting the relevant information that will feed the PROLEG reasoner. These resources are integrated into a natural language processing pipeline able to take a natural language text as an input and return its PROLEG version. Also an ontology, called the Contract Workflow Ontology, is proposed for representing the extracted information in a standard way.

'person A' bought this_real_estate from 'person B' at the price of 200000 dollars by contract0 on 1/January/2018. But 'person A' rescinded contract0 because 'person A' is a minor on 1/March/2018. However, this rescission was made because 'person B' threatened 'person A' on 1/February/2018. It is because 'person B' would like to sell this_real_estate to 'person C' in the higher price. So, 'person A' rescinded rescission of contract0 on 1/April/2018.

```
minor(personA).
agreement_of_purchase_contract(personA,personB,this_real_estate,200000,2018
    year 01 month 01 day,contract0).
manifestation_fact(rescission(contract0),personA,personB,2018 year 03 month 01 day).
fact_of_duress(personB,personA,rescission(contract0),2018 year 02 month 01 day).
manifestation_fact(rescission(rescission(contract0)),personA,personB,2018 year
    04 month 01 day).
```

Fig. 1. Example of an input text and its expected output.

The remain of this paper is as follows. Section 2 presents related work. Section 3 introduces problem and the reasoning system PROLEG, describing the clauses on which the natural language will be translated. In Section 4, an analysis of the main challenges found is done. Section 5 presents how our framework tackled these problems, outlining the different steps and its main functionalities. Section

[3] According to Minsky [19], a frame is *'a data-structure for representing a stereotyped situation'.*

6 presents the outputs of our framework. Finally, Section 7 outlines the main points of our work along with some conclusions and next steps.

2 Related work

Although work on extracting rules in the legal domain has been extensively tackled in literature [8, 9, 21, 27], most efforts focus on regulations and normative text, but not on semi-formal documents dealing with the binding agreements.

The work by Biagioli et al. [3] includes for instance the idea of representing different types of provisions in normative texts as logical structures or frames; nevertheless, these frames output are XML files, not logical clauses, where each provision has some metadata arguments independent to other provisions. On the other hand, Araujo et al. [1] consider a series of legal events in Brazilian Portuguese. Using domain and linguistic knowledge, as well as ontologies, they develop rules for detecting these events via an OWL reasoner. Similarly, Wyner et al. [27] use different NLP tools and resources such as VerbNet [26] to extract events and some related roles from Regulations in English. Other approaches also use ontologies and resources such as WordNet [18] for obtaining semantic information about concepts of interest, such as *obligation, permission* and *prohibition* [8]. Although semantics is a common approach [1, 8, 15, 27], it must be noted that not all proposals rely on NLP or semantics, such as the work by Moulin et al. [21]. Finally, the work by Nakamura et al. [22], later extended to deal with references [14], present a methodology to translate natural language Japanese law texts to logical forms following the Davidsonian Formalization.

The representation of contracts in a digital form has been made in many different forms for different purposes, but none of them matches well enough the representation of PROLEG clauses. The contract itself has been represented in different XML forms, from the well structured OASIS eContracts [16] format to the practical ebXML agreements or the RuleML based business rules [11]. Most efforts to represent the contract in a formal way lean towards defining deontic logic systems, such as Governatori's Business Contract Language intended to address contract violations [10], Daskalopulu's approach to tackle subjective visions on the contract [6] or Prisacariu's effort to consider temporal aspects [23]. However, not many efforts focus on the contract within a workflow, being among the most interesting the Kabilan's ontologies [13] and Molina's [20]. Kabilan identifies at least three perspectives under which a contract can be analyzed: the legal one, the business one, and the information systems one. From each of these perspectives, it is not trivial to abstract a common model for the representation of contracts and contract workflows. Commercial law is different from jurisdiction to jurisdiction, each organization has its own in-house business policies with respect to contract management and information systems are simply too diverse.

When it comes to representing contracts with the purpose of publishing and linking contract information, contract formats are even more scarce. LKIF [12]

devotes a class to Contract[4], but provides no support to contract workflows. FrameNet [2] could be used to represent contracts to some extent (using elements such as *Documents*[5] or *Being_obligated*[6] in FrameNet), but these options do no reflect the information needed for the related PROLEG clauses. Similarly, the *Commerce_buy*[7] frame provides a lot of information on the context of the purchase, but does not consider the contract, focus of our research. The representation of contracts as RDF is not supported by any massively adopted ontology, and there are not many standards or public ontology-based specifications to choose from. One of the possible choices is the Media Contract Ontology [24], ISO standard to support the representation of contracts as RDF but nonetheless domain-specific.

3 The problem

In this section, we will present the problem and introduce the frames developed for representing it, as well as the reasoning system PROLEG. Let us start analyzing the example exposed in Fig. 1.

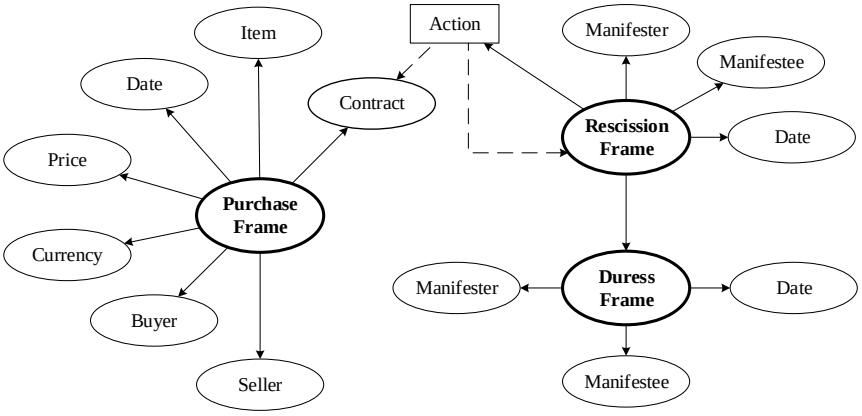


Fig. 2. The three different frames in the framework (Purchase, Rescission and Duress) and how they interact. An action can be a contract or a rescission, therefore a rescission can be of a contract or of another rescission. A duress is also necessarily attached to a rescission.

In the input text, we find different events related to the status of the contract. First, the purchase is uttered via a contract; then, a rescission is claimed,

[4] http://www.estrellaproject.org/lkif-core/norm.owl#Contract
[5] https://framenet2.icsi.berkeley.edu/fnReports/data/frameIndex.xml?frame=Documents
[6] https://framenet2.icsi.berkeley.edu/fnReports/data/frameIndex.xml?frame=Being_obligated
[7] https://framenet2.icsi.berkeley.edu/fnReports/data/frameIndex.xml?frame=Commerce_buy

adducing the fact that one of the parts was a minor. In the third sentence, a fact of duress (threatening) on one of the parts is issued. Additionally, the cause of the ending of the contract is expressed in the following sentence. Finally, in the last sentence the former rescission is rescinded, what makes the contract valid again. It must be noted that we are actually not interested in the information on the fourth sentence, since we just want information about the contract status and the 'real reasons' behind the fact of its rescission are not relevant, but just the fact of it being rescinded at some point of time. For modeling the relevant situations that can involve a contract, we developed three main frames, depicted in Fig. 2. The framework is also able to extract other relevant events to the system, such as if any of the parts involved in the contract is a minor, but most events are related to these three frames. Some examples of events involving these frames are for instance an agreement of a purchase contract, a manifestation of a rescission or the expression of a duress. The expected representation in PROLEG of the facts relevant to the contract status expressed in the example can be found in Fig. 1. With these facts and the contract law information encoded in its rule base (see Fig. 3), the PROLEG system would be able to derive legal consequences of each of the facts, leading to new conclusions such as if the buyer has the right of handling the goods purchased at some concrete point in time or if a contract or a rescission becomes invalid for some reason, such as the existence of duress or some legal incompatibility, such as one of the parts involved being a minor. In Fig. 4 a visualization of this reasoning can be found.

```
right_to_handing_over_the_goods(Buyer,Seller,Object,ContractID)<=
     valid_purchase_contract(Buyer,Seller,Object,Price,Tcontract,ContractID).
valid_purchase_contract(Buyer,Seller,Object,Price,Tcontract,ContractID)<=
     agreement_of_purchase_contract(Buyer,Seller,Object,Price,Tcontract,ContractID).
exception(
     valid_purchase_contract(Buyer,Seller,Object,Price,Tcontract,ContractID),
     rescission_by_minor_buyer(Buyer,Seller,ContractID,Tcontract,Trescission)).
rescission_by_minor_buyer(Buyer,Seller,ContractID,Tcontract,Trescission)<=
    minor(Buyer),
    manifestation(rescission(ContractID),Buyer,Seller,Trescission),
    before_the_day(Tcontract,Trescission).
manifestation(Action,Manifester,Manifestee,Taction)<=
    manifestation_fact(Action,Manifester,Manifestee,Taction).
exception(
    manifestation(Action,Maniester,Manifestee,Taction),
    manifestation_by_duress(Threater,Manifester,Manifestee,Action,Taction,Tduress,Trescission)).
manifestation_by_duress(Threater,Manifester,Manifestee,Action,Taction,Tduress,Trecission)<=
    fact_of_duress(Threater,Manifester,Action,Tduress),
    before_the_day(Tduress,Taction),
    manifestation(rescission(Action),Manifester,Manifestee,Trecission).
```

Fig. 3. Rulebase of PROLEG.

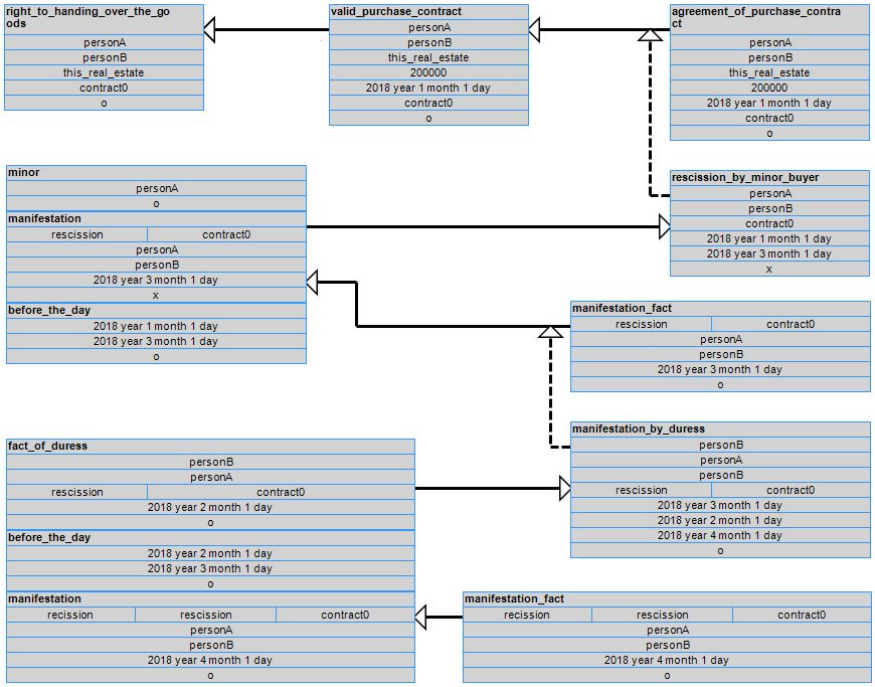


Fig. 4. Visualization of the reasoning done by PROLEG from the facts extracted by our framework.

4 Analysis of challenges

Before explaining our framework, we will expose in this section the different difficulties found during the development of the framework. Each one has a letter assigned, so it can be referred in later paragraphs.

[A] Style of the text Legal texts usually use patterns such as *"A sells L to B by C"*, *"Part A established a contract with Part B"*, or *"'personB' threatened 'person A'"*. In these examples, each of the letters are Named Entities that can be misleading to general NLP tools, that for instance consider *A* as a determiner, changing the whole grammar structure of the sentence. In our case, a preprocessing was done in order to distinguish among real determiners and 'A' parts, and also to eliminate misleading characters (such as ') or blank spaces.

[B] Relevance Differently to other proposals, our aim is not to translate each sentence of the original text, but to extract just relevant facts to the PROLEG system. Therefore, not all the sentences in the text are relevant; in fact, some of them can be actually misleading.

[C] Factuality Besides relevance, some of the sentences in the text processed do not refer to actual facts, but to possibilities, intentions or preferences (e.g., '*A would like to sell a land to B*', '*A preferred to sell it to part D*'). In these cases, some screening should be done in order to prevent these events to enter in our facts base.

[D] Paraphrasing There are many different ways to express the information, both from the syntactical and semantics point of view. While in other domains a lot of semantic resources are available to palliate this phenomena, the Legal Domain presents a very specific terminology uneasy to deal with.

[E] Complexity As already reported in previous literature [7], texts in the legal domain tend to be more complex than texts in other domains. They have higher parse trees, more words per sentence and different POS distribution. These particularities imply an extra challenge when extracting information from them beyond the required preprocessing previously mentioned.

[F] Coreferences and nesting In a natural language text, not all the information of each event is contained in a single sentence. Coreferences are also difficult to handle, especially when there are several manifestations of a type of event (such as a rescission). Also, some information is directly not mentioned and must be inferred using some domain knowledge information. This is the case for instance of a rescission of a contract; if we know that a *contract C* involves *part A* and *part B*, and we know there is a rescission of this contract of *part A* as manifester but it is not explicitly mentioned who is the manifestee, we can assume this role must be *Part B*. Similarly, the duress manifestation of a rescission must be coherent with the information we have of this rescission and the contract it applies to.

[G] Different information For each of the different events processed we require different information. That is an important point that implies some ordering in the rules and preprocessing of the text. An example of this fact is the sentence like "*PersonB rescinded contract because personA was a minor on 1 March 2018.*". In PROLEG clauses, the predicate `minor()` has arity one, so there is no date attached. So even when NLP tools tend to asume that the date mentioned refers to the verb *be* and not to *rescind* (what in fact is linguistically correct and could also be the inference of a human, is an ambiguous sentence), our framework should be able to note the difference.

[H] Matching Since some frames are dependent (e.g., a *duress* must be related to a *rescission*), they must be correctly tracked and matched, so the task is not just about extraction but also about merging.

5 ContractFrames

Our framework ContractFrames makes use of the NLP tool Stanford CoreNLP [17]. We use it for tokenization of the sentences, lemmatization, Part-of-Speech (POS) tagging, Named Entity Recognition (NER) and also for sentence dependence parsing. We also make use of the TokensRegex [4], an annotator that allows setting rules that produce customized annotations. Differently from other options, such as regex, the rules developed in the TokenRegex format are based in previous annotators output, such as POS or NER, being therefore more powerful from the semantic point of view. The steps in our framework are explained below, along with the problems exposed in the previous section that they target:

1. Preprocessing the input text: the first step in our framework deals with problems related to the style of the text and the different information (problems [A] and [G]). The objective of this step is to output a version of the text easier to be understood by the CoreNLP pipeline. To this aim, the following functionalities have been implemented, among others:
   - Algorithm to replace *A*, *'person B'*, or similar misleading expressions for the POS-tagger and the parser.
   - Replacement of appearances of relevant references like 'this_real_state', that often appear in input texts, to standard strings such as *Item1*. These replacements are eventually reversed before producing the final output.
   - Standardization of the dates, that might come as '`dd/Month/yyyy`', a format that the Stanford CoreNLP temporal tagger (SUTime) is not able to detect.
   - Rules to detect the clause with arity one `minor(agent)`, that when found is added to the list of clauses and deleted from the text to avoid misleading parsing as the one explained in the previous section.
2. Annotation with the CoreNLP pipeline: the annotations include tokenization, sentence splitting, POS-tagging, lemmatization, NER (that includes SUTime), parse and the application of our event rules via TokensRegex. The rules developed allows our framework to detect different kind of events

(establishment of contracts, purchases, sales, rescissions, duress...) both in verbal forms (*buy, sell, rescind*) or as noun events (*purchase, sale, rescission*). Each type of event is annotated consequently as a event annotation, so we find the relevant events (problem [B] in the previous section).

3. Parse sentence by sentence: we analyze each sentence separately, assigning one of each of the possible types of frame. Then we analyze the annotations of each token separately:
   (a) If the token has an event annotation, we check if it is negated in the sentence (being therefore not a fact, so we should not transform it into logic) or if it is an intention or a possibility (in this case, we should not either consider it), targeting problem [C]. Once we have verified that it is a fact, we check which type, and then apply different rules to find each its arguments (if available) and express the information as the corresponding frame of the sentence. These rules are mainly applied in the dependency parsing of the sentence, and take into consideration not just the type of the event but also its form (if it is a noun or a verb, if it is active or passive). We cover therefore the different paraphrasing (problem [D]) that can express the information relevant for each frame. Let us analyze for instance the following sentences:

   (1) *landL was sold by PartA to PartB via contractC for 20000 dollars on 13/October/2017.*
   (2) *On 10/13/2017, PartB established a purchase contractC with PartA to buy landL at the price of 20000 dollars.*
   (3) *PartA sold landL to PartB by contractC for 20000 dollars on October 13, 2017.*

   For all of them, the same information is provided, despite of having different words and syntax; the analysis of the dependencies of each sentence is depicted in Fig. 5. Therefore, all these three different input texts imply the same frame (the establishment of a purchase contract) and their PROLEG output should be the following:

```
agreement_of_purchase_contract(partB,partA,landL,20000,2017 year
     10 month 13 day,contractC).
```

   (b) If the token has any other relevant annotation (namely, it has been tagged as a DATE or MONEY by Stanford CoreNLP annotators or as a CONTRACT mention by our rules), we store it as relevant information in the sentence.

   Once each token has been analyzed, we check if there is missing information in frames that have been initialized due to some found event. If so, we complete it with the relevant data we stored (problems [E] and [F]).
4. Once the whole sentence has been processed, we check the information stored in the frames. If there is any information missing, we look for it explicitly (for instance in the values for DATE and MONEY we stored in the previous

step for the current sentence), as well as we also check if new information can complete previous frames. An example of this is the case of a duress: an expression of duress must be linked to a previous rescission, so we check previously mentioned rescissions and link it to the most suitable one (problem [H]).

5. Finally, once all the sentences have been processed, we complete the information on the final frames using some common sense. Let us imagine for instance that we have a rescission with *Manifester A* and *Manifestee B*, with a duress where *B* is the *Manifester* but with no information on the text about the threatened. Since we know the two parties in the contract, we can derive that the *Manifestee* must be *A*.
6. Last step involves just transforming the information in our frames in PROLEG clauses, including reversing the replacement done during the preprocessing step. Since all the relevant information has been stored as our standard frames, translation into PROLEG clauses is straightforward (each frame has one or two clauses whose arguments are among the data in the frame).

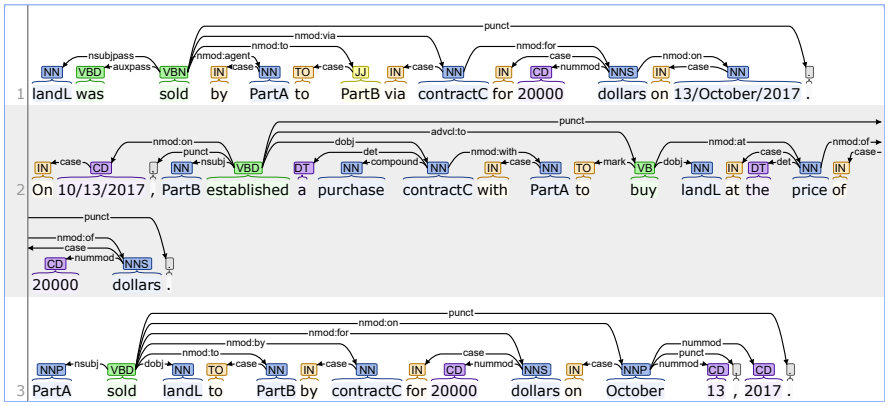


Fig. 5. CoreNLP online demo's[9]output of the enhanced dependencies of the three example sentences where the challenges of paraphrasing become evident. We can see for instance how the buyer (*PartB*) and the seller (*PartA*) play different roles in the sentence depending on the voice (*active/passive*) or the semantics of the verb expressing the purchase frame (*buy/sell/establish a contract*). Also other information can be expressed differently, such as the date.

It must be noted that our framework's NLP pipeline does not involve the corefence annotator from CoreNLP. Although it was initially included in a first version of our framework, we detected that it presented some limitations when dealing with the same event refered both by using nouns and verbs. While it could

[9] http://corenlp.run/

detect that for instance that in *"The rescission of the contract was done on 1 February, 2018. This rescission was cancelled later"* there was a coreference, it did not succeed in cases such as *"A rescinded the contract with B. This rescission was cancelled later"*. We therefore developed the our own algorithm to detect previous potentially similar events and merge them, executed in step 4. Fig. 6 depicts the pipeline of our framework.

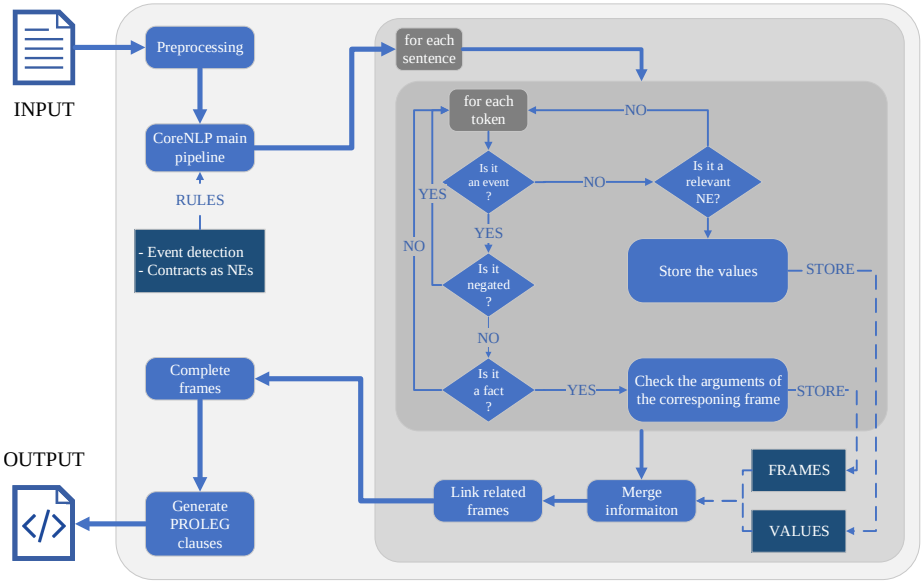


Fig. 6. The pipeline of our framework.

6 Results

The code of ContractFrames[10] is publicly available in a GitHub repository[11]. A dataset with several different texts and their expected input is included in the repository. These texts have been generated by legal researchers, some of them taking no part in the development of the framework. The texts include different types of paraphrasing, both semantic and syntactic, such as the examples depicted in Fig. 5. Also different levels of nesting and events are represented in the dataset, that includes texts of different length explaining the workflow of a contract (the establishment of the contract, the rescission or duress), and even surrounding facts not exploited by the system.

[10] https://mnavasloro.github.io/ContractFrames/
[11] https://github.com/mnavasloro/ContractFrames

The code, written in Java, provides two main classes. The first allows the user to input any text and provides it in the form of PROLEG clauses. The second processes all the files in the dataset, that is also provided and can be extended by the user just by adding new files.

Besides the logic clauses output format, an ontology able to express contract has been developed. This ontology, called Contract Workflow Ontology[12], is capable of representing the different types of events processed, such as agreements and rescission, as well as others in the workflow of a general event such as negotiation. A method for generating an output in the form of triples is provided. Fig. 7 shows the Contract Workflow Ontology.

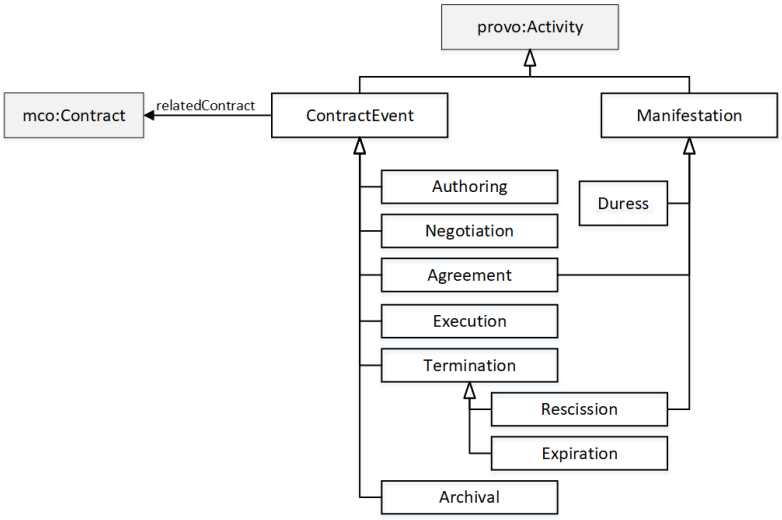


Fig. 7. The Contract Workflow Ontology.

Additionally, the system also generates a .xml output that allows the visualization of the inner custom annotations in the text, namely events and named entities like contracts. An example of this visualization (using the tool GATE [5]) can be found in Fig. 8.

7 Conclusions

In this paper we have presented ContractFrames, a framework to process input text written in natural language including technical legal terminology and produce as output its relevant information in the form of PROLEG logical clauses. The framework is able to recognize several different kind of events, analyze if they are actual facts and extract relevant related information in the form of a

[12] https://mnavasloro.github.io/ContractFrames/datamodel.html

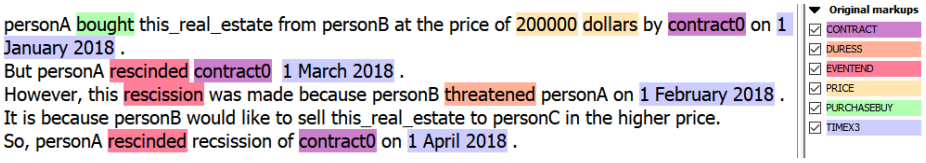


Fig. 8. Visualization of our custom annotations in XML.

frame, deriving omitted information to some extent. Also, an ontology has been created as a data model to store the relevant information of the case and compound the whole contract workflow. This way, any other logical system might also benefit of the processing and the information extracted, and would be able to generate a custom output from this knowledge representation.

Finally, although our framework has been able to successfully process all the example texts produced by legal researchers not involved in its coding, it still has some limitations that will be handled as future work. We therefore have several research lines for improving ContractFrames. For now, the framework is not able to recognize appositions nor naming statements such as *"the contract, entitled from now 'contract0', (...)"* . Despite these expressions are not very common, we consider that exploring new rules for detecting these kind of alternative paraphrasing, eventually to be added to our framework, will be useful. Also more frames able to represent other legal situations and rule sets for populating them will be developed, as well as common sense techniques to derive non explicit information.

References

1. Araujo, D.A., et al.: Automatic information extraction from texts with inference and linguistic knowledge acquisition rules. In: Proceedings of the 2013 IEEE/WIC/ACM International Joint Conferences on Web Intelligence and Intelligent Agent Technologies-Vol. 3. pp. 151–154. IEEE Computer Society (2013)
2. Baker, C., et al.: The berkeley framenet project. In: Proceedings of the 17th international ACL-Vol. 1. pp. 86–90. Association for Computational Linguistics (1998)
3. Biagioli, C., et al.: Automatic semantics extraction in law documents. In: Proceedings of the 10th ICAIL. pp. 133–140. ACM (2005)
4. Chang, A.X., Manning, C.D.: TokensRegex: Defining cascaded regular expressions over tokens. Tech. Rep. CSTR 2014-02, Department of Computer Science, Stanford University (2014)
5. Cunningham, H., et al.: Getting More Out of Biomedical Documents with GATE's Full Lifecycle Open Source Text Analytics. PLOS Computational Biology **9**(2), 1–16 (02 2013)
6. Daskalopulu, A., et al.: Evidence-based electronic contract performance monitoring. Group decision and negotiation **11**(6), 469–485 (2002)
7. Dell'Orletta, et al.: The splet–2012 shared task on dependency parsing of legal texts. In: Semantic Processing of Legal Texts (SPLeT-2012) Workshop Programme. p. 42 (2012)

8. Dragoni, M., et al.: Combining nlp approaches for rule extraction from legal documents. In: 1st Workshop on MIning and REasoning with Legal texts (MIREL 2016) (2016)
9. Francesconi, E.: Legal rules learning based on a semantic model for legislation. In: Proceedings of the LREC 2010 Workshop on the Semantic Processing of Legal Texts (SPLeT-2010). Malta, May 2010. p. 46 (2010)
10. Governatori, G., Milosevic, Z.: A formal analysis of a business contract language. International Journal of Cooperative Information Systems **15**(04), 659–685 (2006)
11. Grosof, B.N.: Representing e-commerce rules via situated courteous logic programs in ruleml. Electronic Commerce Research and Applications **3**(1), 2–20 (2004)
12. Hoekstra, R., Breuker, J., Di Bello, M., Boer, A., et al.: The lkif core ontology of basic legal concepts. LOAIT **321**, 43–63 (2007)
13. Kabilan, V.: Contract workflow model patterns using bpmn. In: Proceedings of the 10th International Workshop on EMMSAD (2005)
14. Kimura, Y., et al.: Treatment of legal sentences including itemized and referential expressions – towards translation into logical forms. In: New Frontiers in Artificial Intelligence. pp. 242–253. Springer Berlin Heidelberg, Berlin, Heidelberg (2009)
15. Kiyavitskaya, N., et al.: Automating the extraction of rights and obligations for regulatory compliance. In: International Conference on Conceptual Modeling. pp. 154–168. Springer (2008)
16. Leff, L., Meyer, P.: econtracts 1.0 committee specification. OASIS LegalXML Technical Report, http://docs.oasis-open.org/legalxmlecontracts (2007)
17. Manning, C.D., et al.: The Stanford CoreNLP Natural Language Processing Toolkit. In: Proceedings of the 52nd Annual Meeting of the ACL 2014, System Demonstrations. pp. 55–60 (2014)
18. Miller, G.A.: Wordnet: a lexical database for english. Communications of the ACM **38**(11), 39–41 (1995)
19. Minsky, M.: A framework for representing knowledge (1975)
20. Molina-Jimenez, C., et al.: Contract representation for run-time monitoring and enforcement. In: IEEE International Conference on E-Commerce. pp. 103–110 (2003)
21. Moulin, B., Rousseau, D.: Automated knowledge acquisition from regulatory texts. IEEE Expert **7**(5), 27–35 (Oct 1992). https://doi.org/10.1109/64.163670
22. Nakamura, M., et al.: Towards translation of legal sentences into logical forms. In: Satoh, K., et al. (eds.) New Frontiers in Artificial Intelligence. pp. 349–362. Springer Berlin Heidelberg, Berlin, Heidelberg (2008)
23. Prisacariu, C., Schneider, G.: A dynamic deontic logic for complex contracts. The Journal of Logic and Algebraic Programming **81**(4), 458–490 (2012)
24. Rodríguez-Doncel, V., et al.: Overview of the mpeg-21 media contract ontology. Semantic Web **7**(3), 311–332 (2016)
25. Satoh, K., et al.: Proleg: An implementation of the presupposed ultimate fact theory of japanese civil code by prolog technology. In: New Frontiers in Artificial Intelligence. pp. 153–164. Springer Berlin Heidelberg, Berlin, Heidelberg (2011)
26. Schuler, K.K.: Verbnet: A broad-coverage, comprehensive verb lexicon (2005)
27. Wyner, A.Z., Peters, W.: On rule extraction from regulations. In: JURIX. vol. 11, pp. 113–122 (2011)

An Interdisciplinary Methodology to Validate Formal Representations of Legal Text Applied to the GDPR

Cesare Bartolini[1], Gabriele Lenzini[1], and Cristiana Santos[2]

[1] University of Luxembourg,
Interdisciplinary Centre for Security, Reliability and Trust (SnT) *
{cesare.bartolini, gabriele.lenzini}@uni.lu
[2] Research Centre for Justice and Governance (JusGov)
School of Law, University of Minho, Portugal
cristiana.teixeirasantos@gmail.com

Abstract. The modelling of a legal text into a machine-processable form, such as a list of logic formulæ, enables a semi-automatic reasoning about legal compliance but might entail some anticipation of legal interpretation in the modelling. The formulæ need therefore to be validated by legal experts, but it is unlikely that they are familiar with the formalism used. This calls for an interdisciplinary validation methodology to ensure that the model is legally coherent with the text it aims to represent but that could also close the communication gap between formal modellers and legal evaluators. This paper discusses such a methodology, providing an human-readable representation that preserves the formulæ's meaning but that presents them in a way that is usable by non-experts. We exemplify the methodology on a use case where Articles of the GDPR are translated in the Reified I/O logic encoded in LegalRuleML.

Keywords: General Data Protection Regulation; GDPR; data protection; legal validation; usability.

1 Introduction

Representing legal text in a machine-readable form is a well-consolidated field of research and industry. Supported by plenty of adequate models and tools [5], it has facilitated knowledge base building, enabled information retrieval processes, and supported decision-making.

Modelling legal text in a machine-readable format is also preparatory to a semi-automatic reasoning about legal compliance. Machines do not deliberate, but the logical steps they follow to assess compliance can increase (or decrease) one's confidence in a legal argumentation. This seems to be an interesting application in relation to the newly-established General Data Protection Regulation (GDPR), where computer scientists and lawyers are debating what measures and practices for data protection are compelled by law.

* Bartolini and Lenzini are supported by the FNR CORE project C16/IS/11333956 "DAPRECO: DAta Protection REgulation COmpliance".

As in other domains, modelling requires an interdisciplinary approach between domain experts and knowledge builders. But in the legal doctrine, modelling text faces additional challenges due to a widely-recognized fact: law is not merely the meaning of the text of its articles but the meaning that emerges from a legal interpretation of the text. Thus, any representation of the meaning convened by a text should be aligned with current interpretative consensus on that meaning as it defined by authoritative groups [7].

This calls for a *validation of any legal knowledge formalization*, and the validation should not be exhausted only within the discipline that does the modelling (computer science in our case). In fact:

(i) although the modeller may not be fully aware, in representing the legal text into a machine-processable format, he or she applies a certain degree of legally-critical interpretation. The modeller has to decide on whether a word or sentence represents a concept that has legal relevance; choose representative names for entities in the model; recognize if the text expresses an obligation, a permission, or a prohibition; and so on;

(ii) even though the interpretation required to encode a legal text into a machine-processable model may have a more restricted scope than a general interpretation of the law, it is unjustified to bury modelling decisions under a formalism that only expert modellers can master;

(iii) although the modeller may be confident of the interpretation he or she gives, concluding whether it is legally correct and complete should be the result of a rigorous process, and not of an individual self-assessment. In their turn, legal experts may want to apply their own qualitative criteria on the anticipated legal interpretations introduced in the model;

Besides, contemporary legal practitioners (such as policy-makers, judges, lawyers, administrators, and legal professionals) can be interested in verifying the results of a formal representation of the law and its applications, e.g., to find evidence in the legally-binding text or look for authoritativeness in knowledge representation [10]. Consequently, any legal knowledge base should be grounded in the reality of a juridical conceptualization of the law [4], promoting a reasonable threshold of reliability and authoritativeness to facilitate relevant applications that would transfer academic research to legal industry.

A legal validation is therefore a significant activity, but performing it over a formal model of a legal text, such as a logic, raises a significant issue: experts in law may not be prepared to fully understand the expressions encoded in the model, a situation that is aggravated if the formalism used is meant to be machine-readable rather than human-understandable. Validating the model requires therefore that the formalization is commonly understandable, and the methodology should be driven by usability considerations.

Promoting accessibility and understandability constitutes a sound practice in general; it is also aligned with the modern introduced codes of ethical conduct

in data science[3] and with the GDPR's principles of algorithmic transparency, fairness, and accountability[4].

Contribution This work discusses a methodology for a legal validation of an existing machine-readable formalization of the GDPR. The input model is a set of XML files —referred to as the Data Protection Regulation Compliance (DAPRECO) knowledge base— containing the GDPR's provisions, formalized in Reified Input/Output (RIO) logic formulæ and embedded in LegalRuleML.

A key element in the methodology is an intermediate representation, supposedly human-readable, of the RIO logic formalization of the GDPR's Articles. This human-readable model is what the experts in law should be able to understand in order to give feedbacks on the consistency and completeness of the translation of the articles in the logic formalization.

The work is still preparatory: after introducing the methodology, it discusses two intermediate representations, together with an analysis of their understandability performed by testers, experts in law, who were involved in a usability experiment. One of the intermediate representations has been unanimously assessed as understandable. It can therefore be a potentially good candidate for a human-readable model to use in the execution of the methodology to collect reliable legal feedbacks on the validation of the formalized GDPR articles.

A full validation of the DAPRECO knowledge base is however left as future work. Many details have to be sorted out first, including improving the scalability of the editing of the human-readable representation, an activity that is currently in part performed automatically, but finalized by hand.

2 Related work

The validation phase of legal modeling by domain legal experts is, to the best of our knowledge, mentioned in the methodologies referring to ontological expert knowledge evaluation.

For example, the Methodology for Modeling Legal Ontologies (MeLOn) [9] was created to build legal ontologies in order to help legal experts model legal concepts, using the principles of data modification. Evaluation parameters consist in: i) completeness of the definition of the legal concepts; ii) correctness of the explicit relationships between legal concepts; iii) coherence of the legal concepts modelisation; iv) applicability to concrete use cases; v) effectiveness for the goals; vi) intuitiveness for the non-legal experts; vii) computational soundness of the logic and reasoning; viii) reusability of the ontology and mapping with other similar ontologies.

An *ad hoc* experimental validation by legal experts of a legal ontology, the Ontology of Professional Judicial Knowledge (OPJK), is described in [6] whereby

[3] `http://www.fatml.org/`.

[4] See GDPR, Art. 5.2 for the principle of accountability, and Artt. 13.2(f) and 14.2(g) for algorithmic transparency.

the expert validation conforms to phases. It includes i) a 48-question questionnaire whereby experts were asked to express their opinion regarding their level of agreement towards the ontology conceptualization and provide suggestions for improvement; and ii) an experimental validation based on a 10-item usability questionnaire, the System Usability Scale (SUS), tailored to evaluate the understanding and acceptance of the contents of the ontology. The adopted ontology evaluation questionnaire could offer rapid feedback and support towards the establishment of relevant agreement, shareability or quality of content measurements in expert-based ontology evaluation.

An evaluation methodology based on Competency Questions (CQs) has been proposed [13] to identify and evaluate the transformation of legal knowledge from a semi-formal form (Semantics Of Business Vocabulary And Rules - Standard English (SBVR-SE)) [8] to a more structured formal representation (OWL 2). It is based on a set of questions with regard to the subject-matter knowledge and predefined answers the ontology is supposed to give. The answers to such CQs could be viewed as requirements of a thus-constructed ontology. Such methodology can be a great tool to enable the cooperation between legal expert and knowledge modeller. Ontology quality criteria are accounted for.

However, there is a fundamental difference between the methodologies proposed to validate legal ontologies and this work. Ontologies are about concepts, data, entities, and a limited set of relations among them; validation of an ontology is therefore inevitably about assessing the qualities of those objects. Formal models for legal compliance, such as the DAPRECO knowledge base, represent a further step, as they model the logical and deontic structure of a legal text, modalities such as time, and constructs like those enabling a defeasible reasoning. The validation should take these elements into account.

Finally, usability is a requirement, since the formulæ must be validated by people who are not expert in logic. Existing validation processes cannot therefore be reused in the present work.

3 Background

The target of the validation proposed in this work is the DAPRECO Knowledge Base, which contains a formalization of the data protection legislation, particularly with respect to the GDPR. The knowledge base is made up of three interconnected main components: 1.) the legal text; 2.) the conceptual model; 3.) the deontic rules. It is meant to be used to provide a semi-automated assistance to the legal expert, all three components need to be machine-readable. For this reason, consolidated standards and reference formats have been used to model each of the three components.

The *legal text* is modelled in Akoma Ntoso[5], which makes it easy to navigate the document, referencing specific portions of text, using ordinary XML parsers.

[5] Currently stored at `https://github.com/guerret/lu.uni.dapreco.parser/blob/master/resources/akn-act-gdpr-full.xml`.

The *conceptual model* is contained in a legal ontology. specifically designed using the Web Ontology Language (OWL) language in an XML serialization.The ontology itself has been developed following the MeLOn methodology, which is based on a glossary and a set of CQs. The output of this work is an ontology of privacy and data protection, called Privacy Ontology (PrOnto) [12,11][6].

Finally, the *deontic rules* of the GDPR are expressed in Reified Input/Output (RIO) logic, which is an extension of Input/Output logic [14] using reification, a technique added to the logic to avoid nested obligations. A full description of the process followed to build the formulæ is described in [2]. This set of RIO formulæ% , their consistency and completeness regarding the legal are the real target of the validation task[7].

To express RIO formulæ in a machine-readable format, LegalRuleML[8] was used in modelling the formulæ. LegalRuleML is an XML markup language and a developing OASIS standard for representing the fine-grained semantic contents of legal texts, which has elements to represent legal content [1]. The formulæ act as a sort of *trait d'union* between the other two components, as they contain references both to ontological elements of the conceptual model and to the textual portions of the legal document expressed in Akoma Ntoso format.

3.1 How the machine-readable text looks like

The objective of the validation are the formulæ regardless of their expressive form (logic or LegalRuleML serialization). As RIO logic is an extension of Input/Output logic, all formulæ are if-then rules in the form (x, y), such that when x is given in input, y is returned in output. When applied to the legal domain, there are three sets to which rules can belong to: C is the set of constitutive norms, which defines when something counts as something else in the domain. Every pair $(x, y) \in C$ reads as "$x \rightarrow y$",as standard first-order logic implications; O and P are respectively the set of obligations and the set of permissions of the normative system. A pair $(x, y) \in O$ reads as "given x, y is obligatory", while a pair $(x, y) \in P$ reads as "given x, y is permitted".

Both the "if" and the "then" part of each formula are composed by a conjunction of predicates. Each predicate is in the form of the predicate name followed by a list of attributes. The name can be a concept belonging to an ontology (e.g., the PrOnto ontology) or it can be a logical operator. For example, $(\mathsf{PrOnto} : \mathsf{PersonalDataProcessing}\, x\, z)$ refers to a concept in the PrOnto ontology and takes two arguments. The predicate alone is incomplete, because it also needs to describe the two predicates used as arguments. If x is a controller and z some personal data of a data subject, an example may be formula 1.

[6] Currently stored at `https://github.com/guerret/lu.uni.dapreco.parser/blob/master/resources/pronto-v8.graphml`.

[7] The formulæ are available at `https://github.com/dapreco/daprecokb`.

[8] `http://ruleml.org/index.html`.

$$((\mathsf{prOnto} : \mathsf{Controller}\,\mathrm{x}) \wedge (\mathsf{prOnto} : \mathsf{DataSubject}\,\mathrm{w}) \wedge$$
$$(\mathsf{prOnto} : \mathsf{PersonalData}\,\mathrm{z}\,\mathrm{w}) \wedge (\mathsf{prOnto} : \mathsf{PersonalDataProcessing}'\,\mathrm{x}\,\mathrm{z})), \quad (1)$$

Furthermore, in RIO logic a predicate can be *reified* to be used as arguments for other predicates. Thus ($\mathsf{prOnto} : \mathsf{PersonalDataProcessing}'\, e_p\, x\, z$) is a new predicate, different from ($\mathsf{prOnto} : \mathsf{PersonalDataProcessing}, x\, z$); it represents the *possibility* that there is a processing of personal data. This allows e_p to be used as argument to another predicate.

In essence, each formula is expressed as a LegalRuleML rule containing two parts: premise (if) and the consequence (then). The predicates (and their arguments) composing both parts are serialized as RuleML atoms (and variables). The example above, with reification added, is serialized as follows:

Listing 1. LegalRuleML representation of formula 1.

```
<ruleml:Exists>
  <ruleml:Var key=":z">z</ruleml:Var>
  <ruleml:Var key=":x">x</ruleml:Var>
  <ruleml:And>
    <ruleml:Atom>
      <ruleml:Rel iri="prOnto:DataSubject"/>
      <ruleml:Var key=":w">w</ruleml:Var>
    </ruleml:Atom>
    <ruleml:Atom>
      <ruleml:Rel iri="prOnto:PersonalData"/>
      <ruleml:Var keyref=":z"/>
      <ruleml:Var keyref=":w"/>
    </ruleml:Atom>
    <ruleml:Atom>
      <ruleml:Rel iri="prOnto:Controller"/>
      <ruleml:Var key=":x">x</ruleml:Var>
      <ruleml:Var keyref=":z"/>
    </ruleml:Atom>
    <ruleml:Atom keyref=":A3">
      <ruleml:Rel iri="prOnto:PersonalDataProcessing"/>
      <ruleml:Var key=":ep">ep</ruleml:Var>
      <ruleml:Var keyref=":x"/>
      <ruleml:Var keyref=":z"/>
    </ruleml:Atom>
  </ruleml:And>
</ruleml:Exists>
```

4 Validation Methodology

From the experience in the DAPRECO project, it turned out that even IT experts required several and repeated explanations by the modeller to understand the formulaæ. Hence the need for a *human-readable* representation of the formulæ, something which preserves the meaning of the machine-readable model while expressing that meaning clearly to non-expert in logic, ontologies, or XML. A few questions emerged: (i) What goal, beside usability, this human-readable representation should achieve? (ii) How to assess that the human-readable representation preserves the meaning of the formulaæ? (iii) How to assess that that

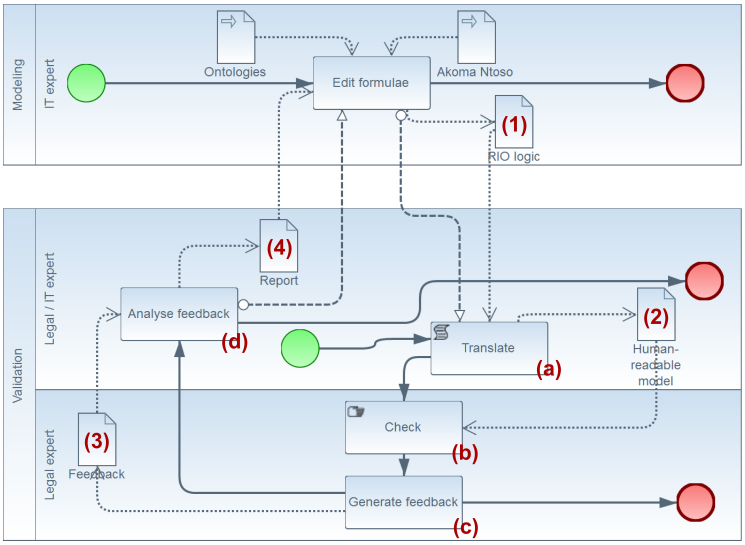


Fig. 1. Workflow of the Modelling and of the Validation methodology.

representation unambiguously and intelligibly conveys that meaning to those not familiar with the modelling?

It also became clear that, to answer these questions, the goal of the validation should be clarified; the human-readable form should be clearly understandable at least with respect to the feedbacks needed from legal evaluators, to verify that the model is done "in the legal right way".

Assuming to have already sorted out what features such a human-readable model must have, the methodology workflow for this work is shown in Figure 1.

The part about the methodology lays on the lower portion of the diagram ("Validation"). The machine-readable version of the modelling of the legal text —in our case, the DAPRECO Knowledge Base— is the output of the modelling effort by the IT expert. That file needs to be processed and rewritten ("Translated", (a) in Figure) into a human-readable representation. The "Human-readable model" (2) is then validated ("Check", (b)) against specific measures telling whether the modelling was correct from a legal point of view. The checking produces a list of "Feedbacks" (3) expressing the assessment of the model's legal qualities, likely in the form of quality measures or answers to a questionnaire. The feedbacks are then analyzed ("Analyze feedback", (d)), e.g., the statistical significance of certain answers will be measured, to compile a "Report" (4) for the IT experts and for the knowledge base builders. The report contains suggestions to review and improve their modelling. This workflow can be iterated until both parties are satisfied.

Due to space constraints, the present work will not delve into the details of each individual step, but only reports on the most critical step in the method-

ology: "Translate". This step generates a human-readable version of the formal representation produced by the "Modelling" phase.

4.1 Translating into a human-readable model

The "Translate" step generates a representation of the formulæ that legal evaluators can use to assess their *legal quality*. Several metrics can be used to measure that quality, such as *completeness* (is all the required domain knowledge explicitly stated, or can it at least be inferred from the vocabulary?); *conciseness* (is there any amount of redundancy in the representation, or is it concise?); *accuracy* (do the deontic modalities represented in the formulæ match the corresponding legal provisions?); *consistency* (is the representation consistent with the law, or does it contain any contradictory knowledge?); and *clarity* (is the representation easy to understand?).

The human-readable representation fed to the legal evaluators to assess legal quality must be easily understandable: the evaluator's mental strain to read the human-readable model to provide legal quality feedbacks should not overcome the effort required to provide feedbacks. A mental effort can be measured [3], but here we propose to measure the understandability of the human-readable representation indirectly as the interrater reliability[9] among a few raters judging whether and how easily they are able to answer a few questions for which they have to understand the meaning of the model.

4.2 Use Case: Translating LegalRuleML of RIO logic formulæ

Our input is the DAPRECO knowledge base, a LegalRuleML file of RIO formulæ supposedly expressing the legal meaning of articles of the GDPR. Reading the knowledge base presents some difficulties, although slightly facilitated by accompanying comments. For instance, in the LegalRuleML serialization, reading the enumerated prohibitions, obligations, reparations, exceptions is not straightforward. According to [15] "the list of [LegalRuleML] elements and their definitions are not sufficient for the consistent and accurate application of the annotations to text, nor is there clarification about how to analyse source text into LegalRuleML. Thus, an annotation methodology is required to connect text to LegalRuleML".

Those issues have to be resolved within the specific validation ("Check") that we foresee in our use case, consisting in the collection of feedback from the evaluators about the questions in Table 1.

To elicit a set of usability requirements for the human-readable model, we performed an internal unstructured inquiry where legal experts were asked to spell out what was making the reading hard and mentally burdensome while trying to answer the previous questions. The inquiry highlighted the following obstacles to a clear understanding of the LegalRuleML of a RIO formula: 1) a formula has little structure, and there are many variables and cross-references

[9] `https://en.wikipedia.org/wiki/Inter-rater_reliability`.

Table 1. The questions that we expect to ask for the "Check" step.

q1	Does the deontic modality of the formula (obligation, permission, constitutive rule) match that of the article?
q2	Does the formula capture all the relevant legal concepts that are expressed in the article (explicit concepts)?
q3	Does the formula capture all other relevant legal concepts not expressed in the article (implicit concepts)?
q4	Does the meaning of the formula completely represent the meaning of the article? What is missing?
q5	Does the meaning of the formula consistently represent the meaning of the article? What is wrong?

between them, forcing the reader to move up and down the code; 2) external references may refer to concepts expressed in the PrOnto ontology, or to logical operators from the RIO logic; 3) the choice of the names of predicates and arguments is not driven by a clear strategy, so that the formula can appear as confusing; 4) whether a formula is an obligation, a permission or an entailment is not immediately readable from its syntax, as it depends on the context, which is defined elsewhere according to LegalRuleML practices; 5) negations are hard to read, as they are structured with two predicates, the first introducing the negation of the second predicate that is expressed positively; 6) RIO logic avoids nesting of obligations and permissions, separating the content of the deontic rule from its bearer in two distinct formulæ. This decision, motivated by the purposes of the logic, can create some confusion, as ultimately there will generally be two separate, and almost identical, formulæ, with the same premises and almost the same consequence.

We address all these problems in a two-step "Translation": the first step is a software that parses the XML, expands and reorders the predicates of the formula; this addresses obstacles 1, 4, 5 and 6. The second is hand-made, to derive an almost natural language break-up version of the formula which, we believe, removes obstacles 2 and 3.

Step One: Automatic Parsing. The output of the automatic translator[10] overcomes the problems enumerated above in the following way: (**i**) variables are substituted with the predicate (taken from PrOnto) that restricts their type; (**ii**) predicates from PrOnto are clearly highlighted in **bold**, whereas predicates from RIO logic and terms that have been introduced for readability's sake are not; (**iii**) the translation of a predicate introduces some terms to try to put everything into context. This technique works quite well due to a good structure of the ontology; (**iv**) the context of a formula (obligation, permission, constitutive) is carried over to the translation; (**v**) negations are treated by translating the predicates in an inline negative sentence. Additionally, when a negation is the object of an obligation, the latter is renamed to a prohibition, and its content

[10] Available at `https://github.com/guerret/lu.uni.dapreco.parser.git`.

expressed positively; (**vi**) if the parser can find another formula with the exact same IF conditions, then they are most likely the content and bearer of an obligation or permission, so the two formulæ are merged into a single translation, which includes both content and bearer.

Article 7.1 of the GDPR can serve as an example: "*Where processing is based on consent, the controller shall be able to demonstrate that the data subject has consented to processing of his or her personal data*"[11].

The (simplified) RIO formula that IT experts wrote (and later encoded in LegalRuleML) to model the provision is shown in formula 2.

$$\begin{aligned}(\,[\,&(\mathsf{RexistAtTime}\ a_1\ t_1) \wedge (\mathsf{and}\ a_1\ e_p\ e_{hc}\ e_{au}\ e_{dp}) \wedge (\mathsf{DataSubject}\ w) \wedge \\ &(\mathsf{PersonalData}\ z\ w) \wedge (\mathsf{Controller}\ y\ z) \wedge (\mathsf{Processor}\ x) \wedge (\mathsf{nominates}'\ e_{dp}\ y\ x) \wedge \\ &(\mathsf{PersonalDataProcessing}'\ e_p\ x\ z) \wedge (\mathsf{Purpose}\ e_{pu}) \wedge (\mathsf{isBasedOn}\ e_p\ e_{pu}) \wedge \\ &(\mathsf{Consent}\ c) \wedge (\mathsf{GiveConsent}'\ e_{hc}\ w\ c) \wedge (\mathsf{AuthorizedBy}'\ e_{au}\ e_{pu}\ c)\,] \rightarrow \\ &[\,(\mathsf{RexistAtTime}\ e_a\ t_1) \wedge (\mathsf{AbleTo}'\ e_a\ y\ e_d) \wedge (\mathsf{Demonstrate}'\ e_d\ y\ e_{hc})\,]\,) \in O \end{aligned} \quad (2)$$

The parser translates the formula as follows:

IF, in at least a situation,

- At time :t_1, the following situation exists:
  - (All of the following (:a_1))
    1. **Processor** (:x) does **PersonalDataProcessing** (:e_p) of **PersonalData** (:z)
    2. **DataSubject** (:w) performs a **GiveConsent** (:e_{hc}) action on **Consent** (:c)
    3. **Purpose** (:e_{pu}) is **AuthorizedBy** (:e_{au}) **Consent** (:c)
    4. **Controller** (:y) **nominates** (:e_{dp}) **Processor** (:x)
- **PersonalData** (:z) is relating to **DataSubject** (:w)
- The **Controller** (:y) is controlling **PersonalData** (:z)
- **PersonalDataProcessing** (:e_p) **isBasedOn Purpose** (:e_{pu})

THEN it must happen that, in at least a situation,

- At time :t_1, **Controller** (:y) is **Obliged** to **AbleTo** (:e_a)
- **Controller** (:y) **Demonstrate** (:e_d) **GiveConsent** (:e_{hc})

Although the translation still requires some mental effort to be processed, it is at least understandable without expertise in logic. The automatic processing also allowed the modeller to verify that the intended meaning has not been changed and is preserved in the translation.

Step Two: Hand Made Break-up. The automatic translation has been further hand-processed. The output is a natural language break-up that highlights the following elements: Premises and the Conclusion of the formula; the Deontic Modality, the Ontological Concepts that can be recognized in the article, Other Ontological Concepts present in the formula but not mentioned in the article; the Contextual meaning, which is what the formula expresses but is not in the article, and the Overall Meaning of the formula. The break-up of Article 7.1 is shown in Table 2.

[11] The full translations for Articles 5.1 and 7.1 can be found in the repository from note 10, in the "jurisin" folder.

Table 2. Structure of the formula's meaning.

Premise	Where processing is based on consent,
Conclusion	the controller shall be able to demonstrate that the data subject has consented to processing of his or her personal data.
Modality	Obligation
Ont. Concepts	Where [Processing] is based on [Consent], the [Controller] shall be [Able to] [Demonstrate] that the [Data subject] [Has consented = GiveConsent] to [Processing] of his or her [Personal data]
Other Ont. Concepts	[Purpose]; [Processor]; [IsAuthorizedBy]; [Nominates]; [IsBasedOn]; [BeAbleTo]
Context	There is a processing, which has a purpose authorized by a consent given by a data subject, and that is what a processor, whom a controller controlling the personal data nominates, does on personal data of the data of the data subject.
Overall Meaning	**Whenever** there is a processing, which has a purpose authorized by a consent given by a data subject, and that is what a processor, whom a controller controlling the personal data nominates, does on personal data of the data of the data subject **then** the controller is **obliged** to able to demonstrate that "data subject gave consent".

4.3 Measuring the usability of the human-readable model

Before collecting the legal experts' feedbacks on the quality of the model, the human-readable model must be read consistently and correctly by the evaluators. The experiment consisted in letting the four legal evaluators (two with knowledge of deontic logic, two without it) answer a few yes/no questions about their understanding of the models of two GDPR provisions, Articles 5.1(a) and 7.1. The input is the human-readable model, but we also fed the original XML formalization and the pre-processed output as control cases, measuring the (pure, not Fleiss Kappa) average interrater agreement between the answers of the evaluators for each model. The questions, built in the wake of the ones used for the validation check in Table 1, were the following. 1. *Can you identify the formula's premise?* 2. *Can you identify the formula's conclusion(s)*? 3. *Can you identify the deontic modality (obligation, permission, other)*? 4. *Can you identify the formula's explicit ontological concepts?* 5. *Can you identify the formula's implicit ontological concepts?* 6. *Do you understand what the formula means?* 7. *Try to rewrite the formula in your own words. Did you succeed?*

We collected the average measure over the two formulæ. The results are shown in Table 3. The hand-processed model is where the evaluators agree almost unanimously over answering 'yes' to all questions, thus indicating high understandability; the control item, the XML file, is where instead there is a majority consensus on being not understandable. Our result also reflects that validators already knowledgeable of logic can somehow read the XML files, despite not fully; unsurprisingly, non-experts could not make any sense of it. Conversely, there is no consensus on the understandability of the automatically-processed model. Supposedly, better usability scores may be attained by training the legal evaluators, but we have not explored this possibility.

Threats to validity We measured understandability as the interrater agreement among a few testers: this measure can suffice to the present goal of having the human-readable model as a candidate within the methodology, but additional

Table 3. Output of the agreement (on 'yes' on 'no') on the readability experiment.

	Commented XML	Intermediate	Human-readable
all	60.6% (no)	40.9%(yes)/59.1%(no)	97.7%(yes)
non-experts	100% (no)	45.5%(yes)/54.5%(no)	95.5%(yes)

measures can provide a deeper evaluation of its usability. More evidence would be needed to assert that our hand-processed model is readable, but since our evaluators generally agree on its understandability, it can already be used to collect the answers that legal experts will give to questions in Table 1 and use them reliably. This will be a next step in the methodology, together with the analysis of the feedback collected during this research.

We stress that, yet, we did not apply the methodology fully: although we collected feedbacks on the legal quality of the formalization of the two GDPR's articles, their analysis is left as future work. Still, the research herein, on the understandability of the human-readable model is a necessary step to later perform the validation methodology.

5 Conclusions and future work

This paper proposed a methodology to perform an interdisciplinary validation of formal representation of a legal text, herein a RIO logic formalization of the GDPR Articles.

Although grounded in a domain-related implementation (the data protection domain), the methodology yields a more general spectrum, since any legal representation calls for a legal validation phase. To the best of the authors' knowledge, a general methodology to evaluate the quality and soundness of a legal representation is novel.

In our case, the generation of a human-readable model was conceived in two steps: an automated translation of the formulæ into a language that unwraps the underlying logic without changing its structure; and a (human-made) post-processing to highlight certain features of the rules that were still hard to detect by legal experts. The final result was then presented to a small group of testers, all from law, who provided feedback.

This work has outlined several interesting issues. First and foremost, inasmuch as human-readable an automated translation can be, it strongly depends on the degree of complexity of the deployed logic. In the use case presented here, for example, the automated translation was generally readable, but some particularly complex formulæ required an added debriefing session, normally not born by pure legal expert; hence the need for a post-processing. The feedback alternated between the correctness of the formulæ and the quality of the translation. The computer expert is therefore required to understand whether a certain feedback requires an improvement of the formula, or if a refined translation (whether a pre- or post-processing) could render a better understanding

by the legal expert. The feedback was delivered through a set of answers to a built questionnaire. The answers from a group of legal experts were combined to discern problems in the translation from those in the formula, or whether there was an interpretative issue (not an error in itself) which could be rather used to improve the knowledge base.

Although confined in its early stages, this work is meant to open a new direction of research in the domain of formal modelling of legal texts, and envisions several follow-ups.

First, the methodology opens the possibility for a thorough validation of the DAPRECO knowledge base, thus improving not only the logic formulæ, but also refining the PrOnto ontology, in a recursive validation process.

Secondly, the results of this approach can also be used to find a clearer and more systematic way to write RIO logic formulæ in the LegalRuleML serialization. In fact, our automated processing highlighted several problematic issues, calling for a process of translation (e.g., in serializing the concepts, in choosing the names of the functions, in adding or omitting variables) that do not depend only on the sheer subjective experience of the modeller. Such a revision will possibly allow achieving better results in the automated translation. If the methodology is streamlined, it might be possible to eliminate the need for manual post-processing, thus leading to a more objective and verifiable human-readable model creation which is scalable to the whole DAPRECO Knowledge Base. Furthermore, a streamlined methodology would allow to embed an interdisciplinary validation process already at the very stage of the modelling of the provisions. This means that while legal provisions are represented (by the computer scientist) into a set of logic formulæ, they would be at one presented to the legal expert to provide feedback. Timely feedback during the creation of the knowledge base could empower the computer scientist to use such expert assessment while modelling subsequent provisions. Surely such a solution would work at its best when the generation of the model from the legal text is made in a semi-automated way, i.e., by means of Natural Language Processing (NLP) techniques, whereby the feedback from the legal expert could be used to improve the processing engine and immediately generate a refined version of the knowledge base. Such an approach can of course be extended to cover different models and domains of modelling, e.g., terms and conditions and privacy polices.

Thirdly, there is a need to define, together with the legal experts, a set of metrics that can express a validator's acceptable assessment on the legal quality of the formalization. In Subsection 4.1, we anticipated a few of possible metrics (i.e., conceptual completeness, conciseness, accuracy, consistency and clarity), but those were not thoroughly investigated yet. This may lead to a revision of the current human-readable model.

References

1. Athan, T., Governatori, G., Palmirani, M., Paschke, A., Wyner, A.: Legalruleml. In: Faber, W., Paschke, A. (eds.) Reasoning Web, Information Systems and Applications, incl. Internet/Web, and HCI, vol. 9203, pp. 151–188. Springer (2015)

2. Bartolini, C., Giurgiu, A., Lenzini, G., Robaldo, L.: Towards legal compliance by correlating standards and laws with a semi-automated methodology. In: BNAIC 2016: Modern Trends in Artificial Intelligence, Communications in Computer and Information Science, vol. 765, pp. 47–62. Springer International Publishing (2017)
3. Benenson, Z., Lenzini, G., Oliveira, D., Parkin, S., Uebelacker, S.: Maybe Poor Johnny Really Cannot Encrypt: The Case for a Complexity Theory for Usable Security. In: Proc. of the 2015 New Security Paradigms Workshop, NSPW 2015, Twente, The Netherlands, September 8-11, 2015. pp. 85–99 (2015)
4. Boella, G., Janssen, M., Hulstijn, J., Humphreys, L., van der Torre, L.: Managing legal interpretation in regulatory compliance. In: Verheij, B., Francesconi, E., Gardner, A. (eds.) Proceedings of the Fourteenth International Conference on Artificial Intelligence and Law (ICAIL). pp. 23–32. ACM (June 2013)
5. Casanovas, P., Palmirani, M., Peroni, S., van Engers, T., Vitali, F.: Semantic web for the legal domain. Semantic Web 7(3), 213–227 (March 2016)
6. Casellas, N.: Ontology evaluation through usability measures. In: On the Move to Meaningful Internet Systems: OTM 2009 Workshops, Lecture Notes in Computer Science, vol. 5872, pp. 594–603. Springer (2009)
7. Francesconi, E.: A learning approach for knowledge acquisition in the legal domain. Approaches to Legal Ontologies, Law, Governance and Technology Series 1, 219–233 (2011)
8. Lévy, F., Nazarenko, A.: Formalization of natural language regulations through SBVR structured english. In: Morgenstern, L., Stefaneas, P., Lévy, F., Wyner, A., Paschke, A. (eds.) Theory, Practice, and Applications of Rules on the Web, Lecture Notes in Computer Science, vol. 8035, pp. 19–33. Springer (2013)
9. Mockus, M., Palmirani, M.: Legal ontology for open government data mashups. In: Parycek, P., Edelmann, N. (eds.) Proceedings of the 7^{th} International Conference for E-Democracy and Open Government (CeDEM). pp. 113–124. IEEE Computer Society (May 2017)
10. Palmirani, M., Cervone, L., Bujor, O., Chiappetta, M.: RAWE: A web editor for rule markup in LegalRuleML. In: CEUR Workshop Proceedings (2013)
11. Palmirani, M., Martoni, M., Rossi, A., Bartolini, C., Robaldo, L.: PrOnto: Privacy ontology for legal compliance. In: Proceedings of the 18^{th} European Conference on Digital Government (ECDG) (October 2018), upcoming.
12. Palmirani, M., Martoni, M., Rossi, A., Bartolini, C., Robaldo, L.: PrOnto: Privacy ontology for legal reasoning. In: Kö, A., Francesconi, E. (eds.) Electronic Government and the Information Systems Perspective, Information Systems and Applications, incl. Internet/Web, and HCI, vol. 11032, pp. 139–152. Springer (2018)
13. Ramakrishna, S., Górski, Ł., Paschke, A.: A dialogue between a lawyer and computer scientist. Applied Artificial Intelligence 30(3), 216–232 (2016)
14. Robaldo, L., Sun, X.: Reified Input/Output logic: Combining Input/Output logic and reification to represent norms coming from existing legislation. Journal of Logic and Computation 27(8), 2471–2503 (December 2017)
15. Wyner, A., Gough, F., Levy, F., Lynch, M., Nazarenko, A.: On annotation of the textual contents of scottish legal instruments. In: Proc. of the 30^{th} Int. Conf. on Legal Knowledge and Information Systems (JURIX). vol. 302, pp. 101–106. IOS Press (December 2017)

AI Risk Assessment Tools: A judge's best friend (?)

A 12-minute read

Eleftherios Chelioudakis[1]

[1] Independent Researcher, currently candidate at the Advanced M.Sc. Digital Humanities, Department of Computer Science, KU Leuven
chelioudakis.ele@protonmail.com

Abstract. In this short paper, the writer deals with the use of Artificial Intelligence Risk Assessment Tools (AI RATs) in the context of the criminal justice system. He describes the different stages of an AI RAT's life cycle and the elements in each of these stages that require special attention. Finally, he concludes that there is the need to carefully regulate, design, study, and implement the AI RATs because the possibility of them to pose a significant danger to our democracy and freedom is particularly high.

Keywords: Risk Assessment Tools, Judiciary, Artificial Intelligence, AI Regulability, Inclusion, Non-discrimination, Fairness, Data quality.

1 Introduction

Table 1. John G. Roberts, Jr., Chief Justice of the United States replies to the question of the Rensselaer Polytechnic Institute President Shirley Ann Jackson during their conversation at Rensselaer, New York on Tuesday, April 11, 2017 [1]

- "Can you foresee a day when smart machines, driven with artificial intelligences, will assist with courtroom fact-finding or, more controversially even, judicial decision-making?"
- "Well, it's a day that's here, and it's putting a significant strain on how the judiciary goes about doing things."

As the quotes in the table above suggest, this short paper deals with the use of Artificial Intelligence (AI) in the judicial system. More than 200 AI Risk Assessment Tools (AI RATs) are used around the world at different settings related to the criminal justice system [2,3]. AI RATs incorporate machine learning algorithms, which generate risk models based on huge volumes of data. They are available at almost every juncture of the criminal justice system, from pretrial risk assessment to prison rehabilitation programs, sentencing proceedings, and parole [4]. In simple terms, AI RATs analyze risks factors and produce a risk assessment classification for an individual. This risk assessment classification will then potentially influence the judiciary in the decision-making process. A decision-making process that AI RATs should not put "a significant strain on" contrary to what John G. Roberts, Jr., Chief Justice of the United States admits.

Famous examples of AI RATs are the "Level of Service Inventory – Revised (LSI-R)" used in Australia [5], Canada [6], and in the U.S.A. [7] and the "Correctional Offender Management Profiling for Alternative Sanctions (COMPAS) used in the U.S.A. [8, 9,10]. Both tools aim to assist the judiciary with sentencing procedures and to identify offenders' risk with regard to recidivism.

Moreover, predictive justice is becoming popular to other parts of the world as well, such as Argentina [11], while international organizations such as the Council of Europe push for a common framework of standards for the use of AI in a court setting [12]. Finally, the European Commission has also expressed its support on investments for the use of AI tools in the field of public administration, including justice [13].

However, justice is not simply just another part of the public administration. Instead, justice has a special position in our societies. It empowers individuals to protect themselves against inequities, and it encompasses core human rights, such as the right to a fair trial, and the right to an effective remedy. Therefore, justice constitutes a core element of our democracy, and matters related to our democracy are matters that should concern every citizen.

Thus, in this short paper, the significant body of literature that exists as regards AI RATs in the judicial system will be used, in order to describe the life cycle of an AI RAT and the elements that require attention during this life cycle. The research goal in this 12-minute read is to answer the question: "What are the different stages of an AI RAT's life cycle, and what are the elements in each of these stages that require special attention?".

2 The first stage: The regulability of the use of AI RATs

The first stage of the AI RAT's life cycle is related to the regulability of the AI RAT. In the following paragraphs of this chapter two themes will be briefly addressed: First, the question as to who is to regulate AI RATs, and second, the question as to what modalities can be used to regulate AI RATs.

The question as to who is to regulate AI RATs is a question of legitimacy, since it refers to who has the rightful entitlement to regulate AI RATs. Such a question is particularly important because private parties are becoming more and more dominant at national and international level and their influence on individuals and governments is rising. Legitimacy is a notion closely related to the notions of legality, moral justifiability, and authority, but it is not identical to them. It is a matter of conformity to rules (regulatory aspect), justification in relation to moral norms and values (morally normative aspect), and consent or representation of those involved or affected (social aspect) [14]. The AI RATs examined in this paper are used in the justice system, and such an environment is completely independent and protected from private interests. Moreover, the justice system of a State is interrelated with this State's independence against both other States and private entities. Therefore, solely national governments can be perceived to have the rightful entitlement to regulate AI RATs used in their justice system.

Regarding the modalities that can be used to regulate AI RATs, there is the need to not lose sight of the focus of this paper, namely AI RATs used in the judicial system.

In general, regulation is a choice among different modalities. Specifically, law, social norms, the market, and the architecture of things which enable or disable certain types of behavior can be used for regulating. Moreover, the regulator is the one to decide which mix of modalities to use in order to achieve the purpose pursued [15]. However, AI RATs used in the criminal justice system assist the judiciary to make decisions which can lead to pretrial detention and/or criminal sanctions. Pretrial detention, criminal sanctions and anything else related to their imposition shall, according to the principle of legality, be provided for by law. Thus, the use of AI RATs as assistance to the judiciary's decisions should be based on an appropriate legal basis which provides for specific safeguards.

3 The second stage: The phase of design and research

The second stage of the AI RAT's life cycle is related to the architecture of the AI RAT and the research on the legal and ethical risks that arise from this technology. In the following paragraphs of this chapter, will be described the elements that require special attention, firstly about the architecture and secondly about the research on the legal and ethical risks that arise.

When it comes to the architecture of the AI RAT, the first element that needs special attention is inclusion in the decision-making process: Decisions on the architecture of the AI RAT should be open to public discussion from the very first stage of the architecture's development. AI RATs are used in justice, and justice is a keystone for our democracies. Therefore, discussions about an AI RAT's design should not take place behind closed doors. Instead it is important for public authorities, civil society organizations, academics, and citizens in general to be able to actively participate in the decision-making process about this architecture. Inclusion creates also a sense of accountability, since when the meetings are not secret their discussions are open to scrutiny. Finally, inclusion creates transparency and boosts trust, since when you can actively participate, pose questions, express opinions, and receive clarifications, you feel involved and have confidence in the decisions agreed upon.

The second element that requires attention lies in the development of AI RATs' architecture: states that are willing to use AI RATs, will need to decide, whether they will develop these tools by themselves, or they will purchase them from a private vendor [16]. If the States decide to utilize a private vendor's option, special attention should be given to trade secrets and Intellectual Property Rights. Since justice is the symbol of fairness, accountability, and transparency, the AI RAT used for its service should be enforcing these principles as well [17]. Therefore, the architecture of the AI RAT shall not be protected by trade secrets because this would interfere with the open discussion process, as described above. Another element that requires attention if States decide to proceed with the private vendor's option, is the access to the personal data used for the risk assessment. Such personal data could be criminal records, racial origin, ethnicity, gender, information on family and personal relations, etc. Thus, it is important to be clarified in the agreement between the State and the private vendor developing the AI RAT, that the vendor shall not be authorized to have access to this data. Instead only

the judiciary, the relevant public authorities, the defendants and their lawyers should be able to access the defendants' personal data used in the risk assessment.

Now, as regards the research on the legal and ethical risks that arise, there are various concerns to be examined. First, attention needs to be paid to the classification of the risk categories of an AI RAT. Using terms like "high risk" to classify individuals' scores on AI RATs can influence the decision-making process of the judiciary and lead to biased choices [18]. Furthermore, the potential of an AI RAT to perpetuate bias might exist in its source code as well, and this is the second concern that requires attention [19]. The quality of the data is of utmost importance for this assessment, because if the AI RAT model is trained on biased data, it will be biased as well. Combating AI RATs' biases, which are based on gender, racial origin, ethnicity, perceived or actual religion, political opinions, sexual preferences and other protected grounds against discrimination should be at the center of this research, in order to correct such faults and biases in the AI RAT.

The correction of such biases and faults is interrelated with the third concern that requires attention, namely the intelligibility of the AI RAT. Even though, intelligibility is hard to achieve, especially with more accurate AI RATs, like neural nets, it is a matter that should be addressed, since it is linked to issues on transparency and fairness [20]. In short, if the AI RAT concludes that an individual constitutes a high-risk, and the reasoning behind this conclusion cannot be explained, then consequently the judiciary's decision based on such conclusion is neither transparent, nor fair. Fairness is the fourth concern that demands attention. Just like intelligibility, the relation of fairness with accuracy is problematic, since maximizing accuracy and fairness at the same time appears to be hard to achieve [21].

This list of concerns is by no means exhaustive, and researchers should be open to other concerns as well, taking into consideration recent technological developments, studies done in related fields, lessons learned from AI RAT's pilot projects, and conclusions reached throughout all the stages of the AI RAT's life cycle.

4 The third stage: Implementing AI RATs

This third stage of the AI RAT's life cycle is related to its implementation. In this phase the AI RAT will be used not on a test set, or on a pilot project like in stage 2, but instead in a real-world context affecting the lives of real people. In the following paragraphs the necessary elements that should accompany the implementation of AI RATs in the criminal justice system will be described.

To begin with, adequate training of the judiciary is of crucial importance. The judges need to be able to recognize the risks which arise from the use of AI RATs, and have a basic understanding of the technology behind such tools. They need to feel comfortable with this technology, so that they will be able to understand its limits. Moreover, the judges should be able to use the AI RATs as a diagnostic tool that can help them to understand the underlying drivers of a criminal activity and not as a predictive tool [22]. Finally, involvement of the judiciary in stage 1 and stage 2 of the AI RATs life cycle is

significant as well. Only then they will be able to follow the AI RATs development and update their knowledge on this matter.

Secondly, during this stage scrutiny over the AI RATs should be obligatory. Here, I do not refer to scrutiny over the judiciary's decisions, but instead scrutiny over the AI RATs classifications and predictions. Specifically, there should be regular and repeated evaluation of the AI RATs validity [23]. A national supervisory authority with independent character, adequate financial resources and a staff of high knowledge and expertise, is the ideal body to be given this supervisory task. Such authority should base its assessments on both quantitative data, such as statistics on the efficiency of the AI RATs, and qualitative data, such as conclusions drawn in a case-by-case basis.

Last but not least, civil society organizations that have at the center of their interest criminal justice should be wide awake. They must act as watchdogs in order to reassure that civil rights are adequately protected. Moreover, they should put pressure on governments and supervisory authorities to properly fulfill their duties.

5 Conclusion

So, is an AI RAT a judge's best friend? By examining the different stages of the AI RAT's life cycle and describing the elements that require special attention in each of these stages, one thing is clearly understood: there is the need to carefully regulate, design, study, and implement such a tool. If the risks that arise from the use of AI RATs in the criminal justice system are not resolved, they will pose a significant danger to our democracy and freedom. Finally, the life cycle of an AI RAT shall not be perceived as a process with a beginning and an end, but instead as a continuing cycle of problem analysis, whereby findings from one stage are fed into another.

References

1. Rensselaer Polytechnic Institute, (12.04.2017). A Conversation with Chief Justice John G. Roberts, Jr. (10:33-10:53). Retrieved from https://www.youtube.com/channel/UCdH4jKoTt9Lr18RxrRqv4gQ (accessed Oct. 2018).
2. Douglas, T., Pugh, J., Singh, I., Savulescu, J. and Fazel, S.. Risk assessment tools in criminal justice and forensic psychiatry: The need for better data. European Psychiatry, 42, pp.134-137 (2017).
3. Singh JP, Desmarais SL, Hurducas C, Arbach-Lucioni K, Condemarin C, Dean K, et al. International perspectives on the practical application of violence risk assessment: a global survey of 44 countries. Int J Forensic Mental Health 2014;13:193–206.
4. Kehl, Danielle, Priscilla Guo, and Samuel Kessler. 2017. Algorithms in the Criminal Justice System: Assessing the Use of Risk Assessments in Sentencing. Responsive Communities Initiative, Berkman Klein Center for Internet & Society, Harvard Law School.
5. Watkins I., The utility of level of service inventory – Revised (LSI-R) Assessments within NSW correctional environments, Corrective Services NSW, Corporate Research, Evaluation and Statistics, Reseach Bulletin No 29, January 2011.

6. Kehl et al., Algorithms in the Criminal Justice System: Assessing the Use of Risk Assessments in Sentencing, supra note 4.
7. Sex Offender Sentencing in Washington State: Predicting Recidivism Based on the LSI-R, Washington State Institute for Public Policy (2006), http:// www.wsipp.wa.gov/ReportFile/935/Wsipp_Predicting-Recidivism-Based-on-the-LSI-R_Predicting-Recidivism-Based-on-the-LSI-R.pdf. (accessed Oct. 2018).
8. California Department of Corrections and Rehabilitation, Correctional Offender Management Profiling for Alternative Sanctions (COMPAS) Fact Sheet, April 2009.
9. Practitioner's Guide to COMPAS Core, Northpointe (Mar. 19, 2015), https://assets.documentcloud.org/documents/2840784/Practitioner-s-Guide-to-COMPAS-Core. Pdf (accessed Oct. 2018).
10. Algorithms in the Criminal Justice System, Electronic Privacy Information Center (n.d.), https://epic.org/ algorithmic-transparency/crim-justice/ (accessed Oct. 2018).
11. Gustavo Corvalán Juan, Artificial Intelligence ,threats, challenges and opportunities Prometea, the first predictive artificial intelligence at the service of justice is Argentinian, Revue Internationale de droit des données et du numérique, Vol.4 (2018).
12. Council of Europe, Parliamentary Assembly (PACE), Recommendation 2102 (2017) 1, Technological convergence, artificial intelligence and human rights.
13. European Commission, Communication from the Commission to the European Parliament, the European Council, the Council, the European Economic and Social Committee and the Committee of the Regions on Artificial Intelligence for Europe, 25.04.2018.
14. Vedder A.H., Towards a defensible conceptualization of the legitimacy of NGOs, January 2007.
15. Lessig L., Code and Other Laws of Cyberspace, 1999.
16. Barabas C., Bavitz C., Budish R., Dinakar K., Dwork C., Gasser U., Hessekiel K., Ito J., Rivest R.L., Virza M., and Zittrain J., An Open Letter to the Members of the Massachusetts Legislature Regarding the Adoption of Actuarial Risk Assessment Tools in the Criminal Justice System, November 2017.
17. Levine D.S., Can We Trust Voting Machines?, October 2012. (accessed Oct. 2018).
18. Barabas et al., An Open Letter to the Members of the Massachusetts Legislature Regarding the Adoption of Actuarial Risk Assessment Tools in the Criminal Justice System, supra note 16.
19. Chouldechova A., Fair prediction with Disparate Impact: A Study of Bias in Recidivism Prediction Instruments, February 2017.
20. Caruana R., Elhadad E., Gehrke J, Koch P., Lou Y., Sturm M., Intelligible Models for HealthCare: Predicting Pneumonia Risk and Hospital 30-day Readmission, August 2015.
21. Berk R., Heidari H., Jabbari S., Kearns M., Roth A., Fairness in Criminal Justice Risk Assessments: The State of the Art, March 2017.
22. Barabas C., Dinakar K., Ito J., Virza M., Zittrain J., Interventions over Predictions: Reframing the Ethical Debate for Actuarial Risk Assessment, Conference on Fairness, Accountability, and Transparency, Proceedings of Machine Learning Research 81:1–15, 2018.
23. The Council of State Governments Justice Center, Risk and Needs Assessment and Race in the Criminal Justice System, May 2016 (accessed Oct. 2018).

COLIEE-2018: Evaluation of the Competition on Case Law Information Extraction and Entailment

Mi-Young Kim[1,2], Yao Lu[2], Juliano Rabelo[2], and Randy Goebel[2,3]

[1]Department of Science, Augustana Faculty, University of Alberta, Camrose, AB, Canada
[2]Alberta Machine Intelligence Institute, University of Alberta, Edmonton, AB, Canada
[3]Department of Computing Science, University of Alberta, Edmonton, AB, Canada
{miyoung2, yao1, rabelo, rgoebel}@ualberta.ca

Abstract. We summarize the evaluation of the case law component of the 5th Competition on Legal Information Extraction/Entailment 2018 (COLIEE-2018). The COLIEE-2018 task includes two tasks in each of statute law and case law. The case law component includes an information retrieval (Task 1), and the confirmation of an entailment relation between an existing case and an unseen case (Task 2). Participation was open to any group based on any approach, and the task attracted 13 teams. We received 6 submissions for Task 1 (for a total of 12 runs), and 4 submissions for the Task 2 (for a total of 8 runs).

Keywords: case law information retrieval, recognizing textual entailment, case law information entailment, AI and law, Juris-informatics

1. Introduction

The Juris-Informatics workshop series was created to promote community discussion on both fundamental and practical issues on legal information processing, with the intention to embrace various disciplines, including law, social sciences, information processing, logic and philosophy, including the exsiting conventional "AI and law" area.

Information extraction and reasoning from legal data is one of the important targets of JURISIN, including legal information representation, relation extraction, textual entailment, summarization, and their applications. Participants in the JURISIN workshops have examined a wide variety of information extraction techniques and environments with a variety of purposes, including retrieval of relevant articles, entity/relation extraction from legal cases, reference extraction from legal cases, finding relevant precedents, summarization of legal cases, and legal question answering.

During the last four years, we held four competitions on legal information extraction/entailment (COLIEE 2014-2017) on a legal data collection, and this helped establish a major experimental effort in the legal information

extraction/retrieval field. We held the fifth competition (COLIEE-2018)[1] this year, with the motivation of continuing to help create a research community of practice for the capture and use of legal information. The previous COLIEE competitions focused on the legal question answering task, based on analysis of Japanese bar law exams and Japanese legal statutes. This year, we have extended the competition to include tasks on case law.

Four tasks are included in the 2018 competition: Tasks 1 and 2 are about case law, and tasks 3 and 4 are about statute law. Here we will introduce the case law competition (Tasks 1 and 2). Task 1 is a legal case retrieval task, and it involves reading a new case Q, and extracting supporting cases S1, S2, …, Sn from the provided case law corpus, hypothesized to support the decision for Q. Task 2 is the legal case entailment task, which involves the identification of a paragraph or paragraphs from existing cases, which entail the decision of a new case. In the next sections, we will describe each task in detail, explain participants' systems, and assessment results.

2. COLIEE Case Law Competition Tasks

COLIEE-2018 data is drawn from an existing collection of predominantly Federal Court of Canada case law, provided by vLex Canada (http://ca.vlex.com). Participants can choose which phase they will apply for, between the two tasks as follows:

Task 1: Legal case retrieval task. Input is an unseen legal case Q, and output is relevant cases in the given legal corpus alleged to support the decision of the input case Q.

Task 2: Confirming an entailment relation between the decision of a new case and a relevant case. Input is a decision paragraph, a short summary and the full contents from an unseen case and a relevant case. Output is selected paragraphs from a relevant case, which entail the decision of the unseen case.

2.1. Task 1: Case law retrieval task

Our goal is to explore and evaluate case law retrieval technologies that are both effective and reliable. The task investigates the performance of systems that search a set of legal cases that support a previously unseen case description. The goal of the task is to accept a query and return noticed cases in the given collection. We say a case is 'noticed' with respect to a query *iff* the case supports the decision of the query case. In this task, the query case does not include a decision, because our goal is to determine how accurately a machine can capture decision-supporting cases for a new case (with no decision).

The process of executing the new query cases over the existing cases and then generating the experimental runs should be entirely automatic. In the training data, each query case is used with a pool of legal cases, and the noticed cases in the

[1] http://www.ualberta.ca/~miyoung2/COLIEE2018/

pool are produced as the answer. In test data, only query cases and a pool of case laws will be included, with no noticed case information.

The format of the COLIEE case law competition data in Task 1 is as follows:

<**pair id="t1-1">**
<**query content_type="summary" description="The summary of the case created by human expert.**">
The parties to this consolidated litigation over the drug at issue brought reciprocal motions, seeking that the opposing party be compelled to provide a further and better affidavit of documents ... (omitted)
</**query**>
<**query content_type="fact" description="The facts of the case created by human expert.**">
[1] Tabib, Prothonotary: The Rules relating to affidavits of documents should be well known by litigants. Yet it seems that parties are either not following them strictly, or are assuming that others are not ... (omitted)
</query>
<**cases_noticed description="The corresponding case id in the candidate cases"**>
18,45,130
</cases_noticed>
<**candidiate_cases description="The candidate cases indexed by id**">
<**candidate_case id="0**"> Case cited by: 2 cases Charest v. Can. (1993)... (omitted)
</candidate_case>
<**candidate_case id="1">** Case cited by: one case Chehade, Re (1994), 83 F.T.R. 154 (TD) ... (omitted)
</candidate_case>
... (omitted)
<**candidate_case id="199">** Desjardins v. Can. (A.G.) (2004), 260 F.T.R. 248 (FC) MLB headnote ... (omitted)
</candidate_case>
</candidate_cases> </pair>

The above is an example of Task 1 training data where query id "t1-1" has 3 noticed cases (IDs: 18, 45, 130) out of 200 candidate cases. The test corpora will not include a <cases_noticed> tag information. Out of the given candidate cases for each query, participants are required to retrieve noticed cases. The candidate cases are made of noticed cases and randomly sampled cases from the database. For those randomly sampled cases, we use vLex Canada's developed algorithm to make sure no relevant cases are identified as sampled cases. Furthermore, two legal experts checked them manually.

2.2. Task 2: Case law entailment task

Our goal in Task 2 is to predict the decision of a new case by entailment from previous relevant cases. As a simpler version of predicting a decision, a decision of a new case and a noticed case will be given as a query. Then a case law textual entailment system must identify which paragraph in the noticed case

entails the decision, by comparing the extracting and compairing the meanings of the query and paragraph.

The task evaluation measures the performance of systems that identify a paragraph that entails the decision of an unseen case. Training data consists of a triple: a query, a noticed case, and a paragraph number of the noticed case by which the decision of the query is alledgedly entailed. The process of executing queries over the noticed cases and generating the experimental runs should be entirely automatic. Test data will include only queries and noticed cases, but no paragraph numbers.

The format of the COLIEE competition data in Task 2 is as following:

<**pair id="t2-1**">
<**query**>
<**case_description content_type="summary" description="The summary of the case created by human expert."**>
The applicant owned and operated the Inn on the Park Hotel and the Holiday Inn in Toronto ... (omitted)
</case_description>
<**case_description content_type="fact" description="The facts of the case created by human expert.">**
... </**case_description**>
<**decision description="The decision of the query case."**> The applicant submits that it is unreasonable to require the applicant to produce the information and documentation referred to in the domestic Requirement Letter within 62 days ... (omitted)
</decision>
<**cases_noticed description="The supporting case of the basic case**">
<**paragraph paragraph_id="1"**>
[1] Carruthers, C.J.P.E.I. : This appeal concerns the right of the Minister of National Revenue to request information from an individual pursuant to the provisions of s. 231.2(1) of the Income Tax Act , S.C. 1970-71-72, c. 63. Background
</paragraph>
<**paragraph paragraph_id="2">**
[2] The appellant, Hubert Pierlot, is the main officer and shareholder of Pierlot Family Farm Ltd. which carries on a farm operation in Green Meadows, Prince Edward Island.
</paragraph>
... (omitted)
<**paragraph paragraph_id="26"**>
[26] I would, therefore, dismiss the appeal. Appeal dismissed. Editor: Steven C. McMinniman/vem [End of document]
</**paragraph**>
</cases_noticed>
</query>
<**entailing_paragraph description="The paragraph id of the entailed case.">13</entailing_paragraph>**
</pair>

The above is an example of Task 2 training data, and the example says that a decision in the query was entailed from the paragraph No. 13 in the given noticed case. The decision in the query does not comprise the whole decision of the case. This is a decision for a portion of the case, and a paragraph that supports the decision should be identified in the given noticed case. The test corpora will not include the <entailing_paragraph> tag information, and participants are required to identify the paragraph number which entails the query decision.

3. Evaluation Metrics and Baselines

The measures for ranking competition participants are intended only to calibrate the set of competition submissions, rather than provide any deep performance measure. The data sets for Tasks 1 and 2 are annotated, so simple information retrieval measures (precision, recall, F-measure, accuracy) can be used to rank each submission. As noted above, the intention is to build a community of practice regarding case law textual entailment, so that the adoption and adaptation of general methods from a variety of fields is considered, and that participants share their approaches, problems, and results.

For Tasks 1 and 2, evaluation measure will be precision, recall and F-measure:

For Task 1:

Precision =(the number of correctly retrieved cases for all queries)/(the number of retrieved cases for all queries),

Recall = (the number of correctly retrieved cases for all queries)/(the number of noticed cases for all queries),

F-measure = (2 x Precision x Recall)/(Precision + Recall).

For Task 2:

Precision =(the number of correctly retrieved paragraphs for all queries)/(the number of retrieved paragraphs for all queries),

Recall = (the number of correctly retrieved paragraphs for all queries)/(the number of relevant paragraphs for all queries),

F-measure = (2 x Precision x Recall)/(Precision + Recall).

For Tasks 1 and 2, we consider the term cosine similarity as the baseline model. Table 1 presents the performances of the baseline model.

Table 1. Baseline performances of Tasks 1 and 2

Tasks	Task 1	Task 2
Precision of Term cosine similarity	0.2649	0.0405
Recall of Term cosine similarity	0.4102	0.5094
F-measure of Term cosine similarity	0.3219	0.0751

4. Submitted Runs and Results

In the overall case law competition, 13 teams registered, 6 teams submitted their system results in Task 1 (for a total of 12 runs), and 4 teams submitted their results in Task 2 (for a total of 8 runs). Some participants submitted multiple runs for a task.

The training data consists of 285 case queries for Task 1, and each query has 200 candidate noticed cases. In the Task 1 training data, each case query has average 9.87 noticed cases. For the task 2 training data, 181 queries have been provided and each query has average 48.59 candidate paragraphs for recognizing entailment relation. In the Task 2 training data, average 1.32 paragraphs have an entailment relation with a query. For the Task 1 test data, 59 case queries and 200 candidate noticed cases for each query were provided where each query has average 10.66 noticed cases. For Task 2 test data, 43 case queries were provided with average 47.19 candidate paragraphs for each query. In the Task 2 test data, average 1.23 paragraphs have an entailment relation with a query.

Tables 2 and 3 summarize a brief description of the submitted systems' techniques. We present the results achieved by runs against the Information Retrieval and Entailment subtasks in Tables 4 and 5, respectively.

Draijer and Verberne (system id: UL) [1] used a Random Forest classifier with eight different features for Task 1. The eight features are More Like This Score on Facts, More Like This Score on Summary, Doc2vec Cosine Similarity distance to Facts, Doc2vec Cosine Similarity distance to Summary, TF-IDF Euclidean distance to Facts, TF-IDF Euclidean distance to Summary, TF-IDF Cosine similarity distance to Facts, and TF-IDF Cosine similarity distance to Summary.

Chen et al. (system id: Smartlaw) [2] proposed using association rules in both Tasks 1 and 2. They first experimented with a machine learning-based model adopting Word2Vec/Doc2Vec as features. But machine learning methods have several disadvantages for this task: first, the tasks have very limited training samples, which make current machine learning models hard to achieve good performance. Second, the space consumption of datasets and the computational cost of training exponentially increase when the size of data expands. To enhance

Table 2. Approaches of submitted systems for Task 1

ID	Run	Approaches
HUKB[4]	HUKB1	Using all case parts, building queries from the summary data and considering all candidates for the database. Returning a variable number of results for each query.
	HUKB2	Using all case parts, building queries from the summary data and considering all candidates for the database. Returning a fixed number of results for each query.
JNLP[3]	JNLP-r=2.5	Combining lexical features and latent features embedding summary properties (parameter range r is 2.5 which determines the interval for selection)
	JNLP-k=10	Combining lexical features and latent features embedding summary properties (the average number of noticed cases is 10)
Smartlaw[2]	Smartlaw	Co-occurrence association model
UA[5]	UA	Pairwise paragraph similarity based features
	UA-postproc	Pairwise paragraph similarity based features with post processing
	UA-smote	Pairwise paragraph similarity based features augmenting the training data with SMOTE
UBIRLED [6]	UBIRLED-1	k-Nearest neighbor search with TFIDF for ranking. Elimination of 75% of the lowest scoring candidate cases
	UBIRLED-2	Ranking with TFIDF then filtering candidates with scores lower than a threshold calculated by the average score of the top 5 documents, divided by 2.
	UBIRLED-3	Ranking with TFIDF and k-Nearest neighbor search, then filtering candidates with scores lower than a threshold calculated by the average score of the top 5 documents, divided by 2.
UL[1]	UL	Random Forest classifier with eight different features

the scalability of the solutions, they propose two association rule models: what is labelled as basic association rule model, and another co-occurrence association rule model. The basic association rule model considers only the similarity between the source document and the target document, and it does not leverage a

Table 3. Approaches of submitted systems for Task 2

ID	Run	Approaches
Smartlaw [2]	Smartlaw	Co-occurrence association model
UA[5]	UA	Similarity based features fed to a Random Forest classifier with 250 estimators
	UA-100	Similarity based features fed to a Random Forest classifier with 100 estimators
	UA-500	Similarity based features fed to a Random Forest classifier with 500 estimators
UBIRLED [6]	UBIRLED-1	Keywords were extracted using Python Keyphrase Extraction toolkit. Then a K-NN search with TF-IDF was used for ranking.
	UBIRLED-2	Facts, Decision, and Paragraphs were mixed together to formulate a query and then UBIRLED-1 approach was used.
	UBIRLED-3	Facts and Summary were mixed together to formulate a query and then UBIRLED-1 approach was used.
UNCC0	UNCC0	Ensemble machine learning with SMOTE resampling technique

manually labeled relevancy dictionary. The co-occurrence association rule model uses a relevancy dictionary in addition to the basic association rule model.

Tran et al. (system id: JNLP) [3] explored benefits from analyzing legal documents' summaries and logical structures for Task 1. They extended the summary of both the query and the candidates to include more attributes from fact/paragraphs. They propose to obtain document embedding information guided by the document summary. This information is used to estimate the phrasal scores for each document given their summary and paragraphs. Subsequently, they train the model with the summary acting as gold catchphrases and paragraphs acting as document sentences. After building the trained model, they generate a latent summary in continuous vector space. For the ranking of candidates, they use two selection strategies: hard top k, and flexible bound relative to score deviation.
UNCC0 applied ensemble learning using the following classifiers: logistic regression, XGBoost classifier, Random forest classifier, and Support Vector Machine classifier. They used resampling of input data using SMOTE for further training.
Yoshioka and Song (system id: HUKB) [4] built an IR system for the task 1 by using the following two steps to retrieve the referred cases: first (1) they build a ranked retrieval, using an IR system to rank candidates. Since the input queries are full text case laws consisting of several parts (summary, citations, paragraph list, etc), they experimented using different parts for building the target database and the queries. They also analyzed the effect of building one database per query (using only the given candidates for that query), and then building one database using all candidates. Their best performance was achieved when the database

used all available case parts; the queries used only the summary and the database was constructed with all candidates. In their second technique (2) from a selection of the referred cases, they choose which of those cases returned in step (1) are going to be used as their system's answer. They tried two strategies: first, select the top *n* ranked cases (*n* fixed a priori), then select a variable number of cases by checking the similarity with non-related cases.

Rabelo et al. (system id: UA) [5] modeled tasks 1 and 2 as binary classification problems. For Task 1, they constructed feature matrices by using a cosine similarity measure between paragraphs from the query case and each candidate case. Those matrices were then transformed into fixed size feature vectors via a histogram approach with pre-determined score bounds, and given to a Random Forest classifier. They also applied post processing to leverage statistical a priori knowledge. Since the dataset in Task 1 is very imbalanced, they under-sampled the dominant class and over-sampled the rarer class by synthesising samples with SMOTE. Their approach for Task 2 was also based on extracting similarity-based features from the query and noticed cases, and feeding those features to a Random Forest classifier.

Lefoane et al. (system id: UBIRLED) [6] propose an approach based on Information Retrieval and unsupervised learning to Task 1: TFIDF is used as a similarity measure between a query and candidate cases. A k-nearest neighbor search with TFIDF as a distance measure is also used. They first rank documents according to their relevance to the query, then apply filtering to exclude the lowest scoring documents from relevant cases, using a threshold value to cut off non-relevant case judgments.

In Table 4, we can see that most systems show better performance than the baseline model. The JNLP system shows the best performance combining lexical features and latent features embedding summary properties (limiting the average number of noticed cases to 10), and it achieved significant increase of the F-measure compared to other systems.

HUKB1 and HUKB2 systems extracted 194 and 191 cases as noticed cases. JNLP-r=2.5 and JNLP-k=10 systems extracted 412 and 399 cases. The Smartlaw system extracted 271 cases, UA, UA-postproc, and UA-smote systems extracted 203, 254, and 247 cases, UBIRLED-1, UBIRLED-2, and UBIRLED-3 systems extracted 392, 453, and 64 cases, and UL system extracted 190 cases. Even though JNLP systems extracted the most cases amongst the systems, they showed the best precision performance. In Task 1, many participants used machine learning classifiers, but the system which used more sophisticated features such as a combination of lexical features and latent features embedding summary properties showed the best performance in this year's competition.

Table 5 reports the results of Task 2, where UA and UA-500 showed the best performance, which is significantly better than the baseline performance. The UA and UA-500 systems used similarity-based features input to a Random Forest

Table 4. IR results(Task 1) on the formal run data

Run	Prec.	Recall	F-m.	Run	Prec.	Recall	F-m.
Baseline	0.2649	0.4102	0.3219	UA-postproc [5]	0.3484	0.4038	0.3741
HUKB1 [4]	0.4974	0.3084	0.3808	UA-smote [5]	0.3539	0.3927	0.3723
HUKB2 [4]	0.4047	0.3037	0.3470	UBIRLED-1 [6]	0.1329	0.6232	0.2191
JNLP-r=2.5[3]	0.5464	0.6550	0.5958	UBIRLED-2 [6]	0.1955	0.7202	0.3075
JNLP-k=10 [3]	**0.6763**	**0.6343**	**0.6546**	UBIRLED-3 [6]	0.5614	0.1017	0.1723
Smartlaw [2]	0.2871	0.4308	0.3446	UL [1]	0.5638	0.3021	0.3934
UA [5]	0.3725	0.3227	0.3458				

Table 5. Entailment results (Task 2) on the formal run data

Run	Prec.	Recall	F-m.	Run	Prec.	Recall	F-m.
Baseline	0.0405	0.5094	0.0751	UBIRLED-1[6]	0.0484	0.8302	0.0914
Smartlaw [2]	0.0465	0.1509	0.0711	UBIRLED-1[6]	0.0495	0.9245	0.0940
UA[5]	**0.2381**	**0.2830**	**0.2586**	UBIRLED-1[6]	0.0467	0.7925	0.0881
UA-100[5]	0.1905	0.2264	0.2069	UNCC0	0.0330	0.0566	0.0417
UA-500[5]	**0.2381**	**0.2830**	**0.2586**				

classifier with different number of estimators. Among the 8 systems, 6 systems showed better performance than the baseline model on Task 2. Task 2 was much difficult than Task 1, and even humans have difficulty in choosing the correct paragraph with the appropriate entailment relations. We can also see the task is difficult based on the low performance on all the systems.

The Tasks 1 and 2 has been newly created in this year's competition, and we think there are many rooms for improvement, such as the evaluation method of Task 2, imbalanced data set, small size set of data which have limitations in applying machine learning techniques, etc. We hope to solve these limitations step-by-step for next competition, to get more robust performances for each Task.

5. Conclusion

We have summarized the results of the COLIEE-2018 competition. Two Tasks were evaluated: (1) Task 1: retrieving noticed cases (information retrieval), and (2) Task 2: extracting paragraphs of relevant case which entail the conclusion of a new case. There were 13 teams who participated in this competition, and we received results from 7 teams. There were 6 submissions to Task 1 (for a total of 12 runs), and 4 submissions to Task 2 (for a total of 8 runs).

A variety of methods were used for Task 1: combining lexical features and latent features embedding summary properties, creating queries from the summaries of cases, and building an information retrieval system to extract noticed cases, co-occurrence association model, pairwise paragraph similarity computation, K-NN, TF-IDF, and a Random forest classifier. Various features were also proposed: features from summary properties, Word2Vec, Doc2Vec, More Like This Score, cosine similarity, Euclidean distance, etc. For Task 2, co-occurrence association model, similarity-based features fed to a random forest classifier, and ensemble machine learning with SMOTE resembling techniques were used. Even though most systems outperformed baseline, all the performances are low, and the task didn't make it easy to identify relevant useful attributes.

For future competitions, we will need to expand the data sets in order to improve the robustness of results. We also need to more deeply investigate how to extract good features for Task 2.

Acknowledgements

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

References

1. Wilco Draijer and Suzan Verberne, “Case law retrieval with doc2vec and Elastic search”, Twelfth International Workshop on Juris-informatics (JURISIN), 2018 (System id: **UL**)
2. Ying Chen, Yilu Zhou, Zhen Lu, Hao Sun and Wenjun Yang, “Legal Information Retrieval by Association Rules”, Twelfth International Workshop on Juris-informatics (JURISIN), 2018 (System id: **Smartlaw**)
3. Vu Tran, Son Truong Nguyen and Minh Le Nguyen, “JNLP Group: Legal Information Retrieval with Summary and Logical Structure Analysis”, Twelfth International Workshop on Juris-informatics (JURISIN), 2018 (System id: **JNLP**)
4. Masaharu Yoshioka and Zihao Song, “HUKB at COLIEE2018 Information Retrieval Task”, Twelfth International Workshop on Juris-informatics (JURISIN), 2018 (System id: **HUKB**)
5. Juliano Rabelo, Mi-Young Kim, Housam Babiker and Randy Goebel, “Legal Information Extraction and Entailment for Statute Law and Case Law”, Twelfth International Workshop on Juris-informatics (JURISIN), 2018 (System id: **UA**)

6. Moemedi Lefoane, Tshepho Koboyatshwene and Lakshmi Narasimhan, “KNN CLUSTERING APPROACH TO LEGAL PRECEDENCE RETRIEVAL”, Twelfth International Workshop on Juris-informatics (JURISIN), 2018 (System id: **UBIRLED**)

Overview of Japanese Statute Law Retrieval and Entailment Task at COLIEE-2018

Masaharu Yoshioka[1,2], Yoshinobu Kano[3], Naoki Kiyota[3], and Ken Satoh[4]

[1] Graduate School of Information Science and Technology, Hokkaido University, N14 W9, Kita-ku, Sapporo-shi, Hokkaido, Japan
yoshioka@ist.hokudai.ac.jp
[2] Global Station for Big Date and Cybersecurity, Global Institution for Collaborative Research and Education, Hokkaido University, Kita-ku, Sapporo-shi, Hokkaido, Japan
[3] Faculty of Informatics, Shizuoka University, 3-5-1 Johoku, Naka-ku, Hamamatsu-shi, Shizuoka 432-8011 Japan
kano@inf.shizuoka.ac.jp, nkiyota@kanolab.net
[4] National Institute of Informatics, 2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo, Japan
ksatoh@nii.ac.jp

Abstract. We present an overview of two tasks (Tasks 3 and 4) of COLIEE-2018 using the Japanese statute law (civil law) and its English translation. Task 3 is the task of retrieving articles to decide the appropriateness of the legal question and Task 4 is the task of entailing whether the legal question is correct or not. There are 17 run submissions from eight teams (including two run submissions from the organizers) for Task 3 and seven run submissions from three teams for Task 4. We present a summary of the evaluation results of both tasks.

Keywords: Legal Information Retrieval · Legal Information Entailment · Query Analysis · Natural Language Processing.

1 Introduction

Competition on legal information extraction/entailment (COLIEE) is a series of evaluation campaigns to discuss the state of the art for information retrieval and entailment using legal texts [7, 6, 4]. In the previous COLIEE (COLIEE 2015-2017), there were two tasks (information retrieval (IR) and entailment) of using the Japanese statute law (civil law). In COLIEE-2018, we conduct two new tasks (IR and entailment) of using a Canadian case law and two tasks of using the Japanese statute law that had the same settings as in the previous campaigns.

For the IR task (Task 3), based on the discussion about the analysis of IR tasks of the previous COLIEE [12], we modified the evaluation measure of the final results and also asked the participants to submit ranked relevant article results to discuss the detailed difficulty of the questions.

For the entailment task (Task 4), we performed categorized analyses to show different issues of the problems and characteristics of the submissions, in addition to the accuracy evaluation, similar to tasks of the previous COLIEE.

This paper summarizes an overview of Tasks 3 and 4 of COLIEE-2018. There were 17 run submissions from eight teams (including two run submissions from the organizers) for Task 3 and seven run submissions from three teams for Task 4. We present a summary of the evaluation results of the two tasks. Sections 2 and 3 discuss the evaluation for Tasks 3 and 4, respectively. Section 4 summarizes the paper with a discussion about the future direction of these tasks based on the evaluation results.

2 Task 3: Japanese Statute Law Retrieval Task

2.1 Task description

Task 3 is the task of retrieving articles to decide on the appropriateness of the legal question. The training and test data of the legal questions were collected from the Japanese bar exam. All the questions and the Japanese civil law articles (1056 articles in total) were provided in two languages, Japanese and English. The English version of the law articles and questions was provided by the organizers. The participants were asked to submit relevant articles for the questions using Japanese or English data. Each participant can submit at most three runs for Task 3.

2.2 Data set

In this evaluation campaign, for this task, questions related to Japanese civil law were selected from the Japanese bar exam. The organizers provided a data set used for previous campaigns [7, 6, 4] as training data (651 questions) and new questions selected from the 2017 bar exam as test data (69 questions).

As the system mostly returns only one article for each question, the number of relevant article(s) for the question affects the system performance. The number of questions classified by the number of relevant articles is listed in Table 1.

Table 1. Number of questions classified by number of relevant articles

number of relevant article(s)	1	2	3	total
number of questions	51	16	2	69

2.3 Submitted runs

The following eight teams (in alphabetical order except for the organizers' team as baseline) submitted their results (17 runs in total). Three teams (HUKB, JNLP, and UA) had experience in submitting results in the previous campaign and four teams (Smartlaw, SPABS, UB, and UE) were new to the campaign.

HUKB (two runs) [13] used structural analysis results (condition, decision) of the article and questions and used Indri [10] to calculate similarity measures among different parts. The SVM-rank [2] software was used to aggregate such similarity measures. HUKB1 decided on the number of returned articles based on the analysis of retrieval difficulty for IR. HUKB2 returned only one article for each question.

JNLP (two runs) [11] used the structural analysis results (requisite and effectuation) of articles and the TF-IDF-based vector space model for calculating the similarity among them. JNLP1 used the similarity between query and articles only for article ranking. JNLP2 calculated the final similarity value as a linear combination of similarity used for JNLP1 and the similarity between the query and the article effectuation part. Both runs returned two articles for all questions based on the analysis of training data.

Smartlaw (three runs) [1] calculated the similarity of a question and an article by checking the similarity between (1–4) gram sets extracted from the question and the article. Based on the experimental analysis, they submitted three runs whose settings for constructing (1–4) gram sets were different; Smartlaw, Smartlaw_2gram, and Smartlaw_3gram used bigram+trigram, and bigram and trigram, respectively.

SPABS (three runs) used recurrent neural network (RNN) models to calculate the similarity between question and articles. For training word embedding, they used English legal documents with Word2Vec. SPABS_bm25 is their baseline result using BM25.

UA (one run) [5] used the same system of COLIEE-2017 for Task 3. This system uses the TF-IDF model of Lucene. [5]

UB (three runs) used the Terrier 4.2 [6] with the PL2 term-weighting model as the IR platform. UB3 used TagCrowd [7] to select important keywords from each question and used them as a query of the IR platform. UB2 used query expansion after UB3 retrieval, and UB1 used word embeddings.

UE (one run) used the rule-based method to retrieve relevant documents.

ORG (two runs) used Indri [10] with simple settings (the question was used as query and all articles with title were indexed as a document) [12].

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

2.4 Evaluation of the Submitted Runs

Table 2 shows the evaluation results of submitted runs including the organizers runs. The official evaluation measures used in this task were F2 measure, precision (Prec.), and recall (Rec.). The terms "ret.", and "rel" represent the number

[5] https://lucene.apache.org/
[6] http://terrier.org/
[7] https://tagcrowd.com/

of returned articles and the number of returned relevant articles, respectively. The columns after the mean average precision (MAP) are explained below.

$$precision = \frac{number\ of\ retrieved\ relevant\ articles}{number\ of\ returned\ articles} \tag{1}$$

$$recall = \frac{number\ of\ retrieved\ relevant\ articles}{number\ of\ relevant\ articles} \tag{2}$$

$$f2 = \frac{5 \times precision \times recall}{4 \times precision + recall} \tag{3}$$

There are two differences in the evaluation measure used for the task compared with the former campaigns.

1. F2 measure was used instead of F measure (harmonic means of precision and recall).
   F2 measure is a variation of F-measure that weights recall higher than precision. If we assume that the IR task is a preprocess to provide relevant article(s) to the entailment system, a set of candidate article(s) including relevant article(s) to the entailment system needs to be provided. [8]
2. Macro average is used instead of micro average.
   In the former campaigns, the micro average (the average of evaluation measures is calculated based on the aggregated numbers of relevant articles, returned articles, and returned relevant articles for all questions) was used for evaluation. However, this measure is not very appropriate for different numbers of relevant articles. For example, for analyzing a recall, questions with multiple relevant articles are more important than one with one relevant article. In addition, when the system returns many articles for one query because of the uncertainty of the returned results, this seriously deteriorates the precision of the micro average. However, using the macro average (each evaluation measure is calculated based on the number of relevant articles, returned articles, and returned relevant articles for each question; after calculating the evaluation measure for each question, the average of such a measure over all questions is calculated), we can reduce the effect of such different characteristics among all retrieved results.

In the previous campaigns, because most of the teams submitted only one or two articles for each question, we could only evaluate the topic difficulties based on the number of systems that can return such articles as the relevant one. However, it is almost impossible to estimate the reason for the problem. For example, some questions have difficulties ranking the relevant articles higher because of vocabulary mismatch, and some questions have difficulties selecting the appropriate one from similar articles (relevant articles are ranked higher but not first-ranked). Therefore, we decided to ask the participants to submit a long ranking list (100 articles) in addition to the selected relevant article candidate list.

[8] This assumption is based on the concept that an entailment system can identify the most relevant part of texts from the provided texts.

This list provides information that could discuss the type of difficulties in retrieving relevant articles. For the long list, the MAP, recall at k (R_k: recall calculated by using the top k ranked documents as returned documents) are used for evaluation measure.

Table 2 shows information about the evaluation measure for the long rank list. However, because UE did not submit this long list, values are given as "-".

Table 2. Evaluation results of submitted runs (Task 3) and the corresponding organizers' run

runid	lang	ret.	rel.	F2	Prec.	Rec.	MAP	R_5	R_{10}	R_{30}
UB3	E	69	54	**0.6964**	**0.7826**	0.6860	**0.7988**	0.7978	0.8539	**0.9551**
UA	E	69	50	0.6602	0.7246	0.6522	0.7451	0.7303	0.7528	0.8539
ORGE1	E	69	49	0.6368	0.7101	0.6280	0.7381	0.7528	0.8090	0.8989
UB2	E	69	47	0.6232	0.6812	0.6159	0.7542	**0.7978**	**0.8652**	**0.9551**
JNLP1	E	138	57	0.6118	0.4130	**0.7126**	0.7398	0.7640	0.8202	0.9213
Smartlaw	E	138	57	0.6042	0.4130	0.7005	0.7036	0.7079	0.7640	0.8315
JNLP2	E	138	56	0.5997	0.4058	0.6981	0.7296	0.7528	0.8090	0.9101
SPABS_bm25	E	138	55	0.5821	0.3986	0.6739	0.7070	0.7753	0.8202	0.9101
UE	E	69	34	0.4516	0.4928	0.4469	-	-	-	-
Smartlaw_3gram	E	69	34	0.4387	0.4928	0.4324	0.4700	0.4494	0.4607	0.5056
UB1	E	69	31	0.4171	0.4493	0.4130	0.5355	0.5730	0.7191	0.8202
Smartlaw_2gram	E	141	34	0.3421	0.3023	0.4275	0.4594	0.4382	0.4831	0.5169
SPABS_rnnen	E	138	19	0.2150	0.1377	0.2536	0.2638	0.3371	0.4494	0.5730
SPABS_rnnsq	E	138	17	0.1957	0.1232	0.2319	0.2662	0.3483	0.4494	0.6067
HUKB2	J	69	53	0.6859	0.7681	0.6763	0.7805	0.7865	0.8427	0.9326
HUKB1	J	74	53	0.6826	0.7536	0.6763	0.7805	0.7865	0.8427	0.9326
ORGJ1	J	69	51	0.6633	0.7391	0.6546	0.7703	0.7753	0.8427	0.9326

Based on the comparison between ORGJ1 and ORGE1, we confirmed that there was no large difference between the English and the Japanese data.

As the average of the relevant articles per query was 1.29 (89/69), the performance of the systems that returned two articles for each question was worse than those that returned one article only. The best-performing system was UB3, which used a tag cloud algorithm to select appropriate keywords to construct the query and the Terrier IR platform to retrieve the final results. Teams that participated in the previous campaigns had almost similar scores except for JNLP, which returned two articles for each question. The performance of new teams, except for UB, was worse than the baseline system.

We discuss the difficulty of the questions based on the averaged evaluation measure among team's top run results for each language (eight results; HUKB2, JNLP1, SPABS_bm25, UB3, UA, Smartlaw, ORGJ, and ORGE). For the questions that have one relevant article, 28 out of 51 questions had an average MAP of 1.0. This means that these questions were easy and no system made the mistake of ranking relevant articles as the first article. For these questions, the

system that returned two articles for each question had a poor precision score (0.5) even though the systems ranked the relevant article as first-ranked article. We will not discuss here the easy questions in detail, and instead only focus on the difficult questions.

Figure 1 shows averages of MAP, R_5, R_{10} for the 23 questions with a single relevant article. In most cases, all of the system could find the relevant articles as higher ranked articles (14 questions had $R_5 = 1$ and two questions had $R_5 = 0.875$, which means that only one system cannot rank the articles in the top five.) There were a few questions for which it was difficult to rank relevant articles higher. The most difficult problem for which nobody managed to rank relevant articles in the top 10 is H29-1-0.

> H29-1-O: A is a nineteen-year-old male and subject to the parental authority of his parents. In the case where A concluded a contract not in writing to the effect that A would make B a gift of 1,000,000 yen from the property which A had obtained by inheritance, even if A revoked the gift on the ground that it was a gift not in writing, if A did not obtain the consent of the person who had parental authority in relation to him with respect to the revocation of the gift, A may rescind the revocation of the gift.

The relevant article (Article 5) uses the keywords "statutory agent" and "parental authority" may serve as "statutory agent." However, the system tends to retrieve articles that discuss "parental authority."

For these questions, the RNN-based approach SPABS_rnnsq selected this article as a third relevant article. Although the overall performance of the RNN-based system is not very effective, this type of system may be good at finding relevant articles with a large vocabulary mismatch.

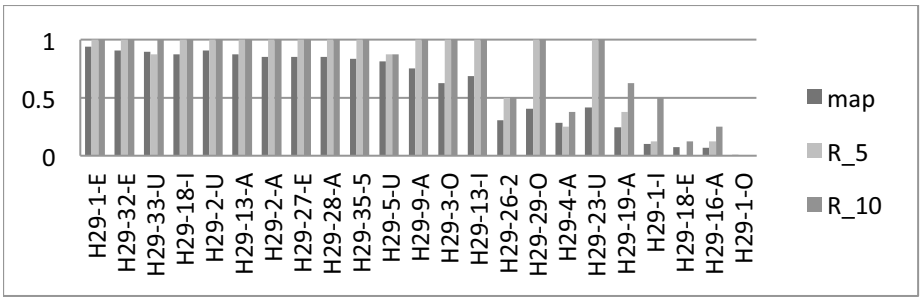


Fig. 1. Averages of MAP, R_5, R_{10} for difficult questions with a single relevant article

Figure 2 shows the averages of precision, recall, MAP, R_5, R_{10} for questions with multiple relevant articles (two questions, H29-28-E and H29-35-I, have three relevant articles and 16 other questions have two relevant articles). There are a few questions where both the first- and second-ranked articles are relevant

(MAP = 1). In other cases, there are many questions for which the content is similar to one of the relevant articles. but the other relevant articles are not so similar to the questions. H29-9-U is one such example of this type of question.

> H29-9-U: If the principal movable owned by A and the accessory movable owned by B are so joined to each other that they can no longer be separated without damaging the same, ownership of the composite Thing shall vest in A and B may demand compensation against A.

Relevant articles for this question are Articles 243 and 248. Article 243 shares the phrase "are so joined to each other that they can no longer be separated without damaging the same, ownership of the composite Thing shall vest in" and it is easy to rank it first. In contrast, article 248 only contains the phrase "may demand compensation." As a result, the content-based similarity is comparatively lower than for the first-ranked article and other articles similar to 243. One of the clues that can be used to identify that Article 248 is relevant is that it has a reference part to Article 243.

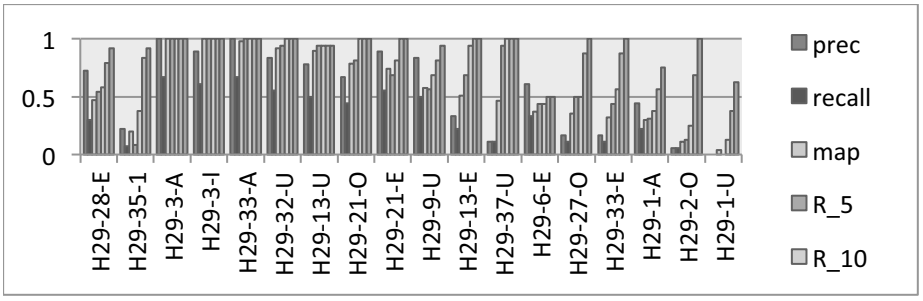


Fig. 2. Averages of precision, recall, MAP, R_5, R_10 for noneasy questions with a single relevant article

2.5 Discussion

As we have conducted a series of campaigns to retrieve relevant articles to entail the questions of the Japanese bar exam, most of the systems succeed in retrieving relevant articles of the simple questions that have only one relevant article and higher vocabulary (phrase) overlap between the question and the relevant article. However, the retrieval performance of the questions with vocabulary mismatch is insufficient. A semantic matching technique including the RNN approach may be one way to tackle this type of problem. However, to avoid the side effect of degrading the retrieval performance of easy questions, it is better to have a preprocess to select whether it is necessary to use such a semantic matching technique.

For the questions with multiple answers, there are many questions for which content-based similarity is inadequate in identifying the second or third supplemental relevant articles. Information about the relationships between articles may be a candidate information resource that is currently underused; further discussion is necessary to tackle this type of problem.

3 Task4: Entailment Task

3.1 Task description

Task 4 is a task to determine entailment relationships between a given problem sentence and an article sentence. Participants should answer yes or no regarding the given problem sentence. Pure entailment tasks were held until COLIEE-2016, where t1 (relevant article sentences) and t2 (problem sentences) were given. However, because of the limited number of available problems, COLIEE-2017 and -2018 did not hold this type of task. In Task 4 of COLIEE-2018, t1 (the relevant articles) was not given, and participants should find the relevant articles by themselves. Training and test data of the legal questions were collected from the Japanese bar exam, as for Task 3. All questions and Japanese civil law articles (1044 articles in total) were provided in two languages, Japanese and English. The English version of the law articles and questions was provided by the organizers. The participants were asked to submit yes/no answers for each question using Japanese or English data.

3.2 Data set

Our data set was derived from the civil law short answer (multiple choice) part of the Japanese legal bar exam. The organizers provided a data set of 644 questions used in the previous COLIEE tasks [7, 6, 4] as training data, and 69 new questions were selected from the 2017 legal bar exam as test data.

3.3 Submitted runs

The following three teams submitted their results. As one team submitted five runs, there were seven runs in total. Two teams (KIS and UA) had experience in submitting results in previous tasks and one team (UE) was new to the task.

KIS (5 runs) [8] analyzed Japanese sentences linguistically, and used predicate argument structures to determine similarities. [9] used frame information to calculate similarity between predicates. Their final results were an ensemble of these different modules by SVM.

UA (one run) [3] used almost the same system as in COLIEE-2017 for Task 4. Their system used condition/conclusion/exception detection rules, and negation dictionaries are created manually.

UE (one run) combined deep neural network with additional features, and word2vec to retrieve the corresponding civil law articles.

3.4 Evaluation of the submitted runs

Table 3 shows the evaluation results of submitted runs. The official evaluation measure used in this task was accuracy. LANG shows the language of the data, J for Japanese and E for English. Correct answers are the numbers of correct system outputs among 69 questions. The baseline system was simply No answers to all questions.

Table 3. Evaluation results of submitted runs (Task 4) and baseline result

Team	LANG	Correct Answers	Accuracy
BaseLine	N/A	35 (All No)	0.5072
UA	?	44	0.6377
KIS_Frame	J	39	0.5652
KIS_mo3	J	38	0.5507
KIS_dict	J	37	0.5362
KIS_SVM	J	36	0.5217
KIS_Frame2	J	35	0.5072
UE	E	33	0.4783

The best system was UA, with an accuracy of 0.6377. The baseline was almost 0.5, because this task was a binary classification, with 35/69 questions being No.

The effect of language differences was unclear. In our statute law tasks, the Japanese legal bar exam was the original data, which was manually translated into English. Team UA used the translation system and the Korean parser internally. The translation process might have absorbed ambiguities and paraphrases.

Because an entailment task is essentially a complex composition of different subtasks, we manually categorized our test data into categories, depending on what sort of technical issues had to be resolved. Table 4 shows our categorization results. As this was a compositional task, overlap between categories was allowed. Our categorization was based on the original Japanese version of the legal bar exam.

3.5 Discussion

Our categorization shown in the previous section suggested several issues and analyses. The largest number among these categories was for the conditions. UA, the best team, was better in this condition category. Their condition detection should have performed successfully. KIS_Frame2, which used the frame information, was good in case roles, person relations, and person roles. Their frame relation would have a certain effect in these deep semantic issues.

Because the distribution of Yes/No answers is quite diverse between submissions, an ensemble could perform better results if we could capture meaningful information for each submission.

Table 4. Technical category statistics of questions, and correct answers of submitted runs for each category. # stands for number of corresponding questions, team names stand for their number of correct answers for the corresponding category, acc stands for accuracy of its left-hand-side correct answers*.

Category	#	UA	acc	UE	acc	KIS1	acc	KIS2	acc	KIS3	acc	KIS4	acc	KIS5	acc
Itemized	3	1	0.33	2	0.67	1	0.33	1	0.33	1	0.33	1	0.33	2	0.67
Numerical priority	3	2	0.67	2	0.67	1	0.33	1	0.33	2	0.67	1	0.33	2	0.67
Entailment	5	2	0.40	2	0.40	1	0.20	1	0.20	4	0.80	2	0.40	2	0.40
Dependency	5	3	0.60	1	0.20	2	0.40	2	0.40	3	0.60	0	0.00	4	0.80
Article search	5	3	0.60	2	0.40	3	0.60	3	0.60	1	0.20	1	0.20	4	0.80
Paraphrase	5	2	0.40	4	0.80	3	0.60	3	0.60	2	0.40	3	0.60	3	0.60
Negation	7	5	0.71	3	0.43	5	0.71	5	0.71	2	0.29	1	0.14	7	1.00
Legal terms	7	4	0.57	2	0.29	2	0.29	2	0.29	3	0.43	4	0.57	3	0.43
Normal terms	9	5	0.56	5	0.56	4	0.44	4	0.44	5	0.56	6	0.67	4	0.44
Predicate argument	9	8	0.89	3	0.33	5	0.56	5	0.56	5	0.56	4	0.44	5	0.56
Verb paraphrase	13	7	0.54	6	0.46	7	0.54	7	0.54	7	0.54	7	0.54	4	0.31
Case role	15	8	0.53	6	0.40	9	0.60	9	0.60	6	0.40	11	0.73	6	0.40
Ambiguity	17	9	0.53	7	0.41	8	0.47	8	0.47	8	0.47	10	0.59	9	0.53
Anaphora	20	13	0.65	5	0.25	12	0.60	11	0.55	8	0.40	8	0.40	13	0.65
Morpheme	25	18	0.72	16	0.64	20	0.80	19	0.76	10	0.40	16	0.64	16	0.64
Person relationship	26	14	0.54	11	0.42	13	0.50	13	0.50	13	0.50	18	0.69	10	0.38
Person role	27	16	0.59	12	0.44	14	0.52	14	0.52	14	0.52	18	0.67	13	0.48
Conditions	31	19	0.61	9	0.29	13	0.42	12	0.39	16	0.52	11	0.35	16	0.52

* KIS1, KIS2, KIS3, KIS4, and KIS5 corresponds to KIS_mo3, KIS_dict, KIS_SVM, KIS_Frame2, and KIS_Frame respectively.

We did not have any end-to-end machine learning system this year, but we had many such systems previously. It would still be difficult to solve these entailment-based question-answering problems. The number of available questions was still too small to perform an effectively supervised machine learning. Evaluations of the training data (past years' problems) showed quite different scores between the different years. As our category analysis shows, many different issues remain to solve these questions; each category would require hundreds or thousands of training data sets at least. Furthermore, such deep semantic issues are still unresolved in general. A better linguistic analyzer would be needed as a base, not just end-to-end systems. In other words, our problems are very good material for challenging and evaluating deep semantics that are still unresolved.

4 Summary

This paper summarizes an overview of Tasks 3 and 4 of COLIEE-2018. For Task 3, we found that there were three types of problem in the test data; i.e., easy questions, difficult questions with vocabulary mismatch, and questions with multiple answers. Most submission systems are good at retrieving relevant answers for easy questions, but it is still difficult to retrieve relevant articles with other question types. It may be necessary to focus on such question types to improve the overall performance of the IR system. For Task 4, the overall performance of the submissions was still not sufficient for their systems to be used in the real application. However, a detailed analysis could capture the characteristics of the submitted systems. We found that to discuss and develop deep semantic analysis issues in the real application, and natural language processing in general is still a challenging task.

Acknowledgement

We would like to thank other COLIEE organizers for their supports to construct training data sets and organize this campaign. This work was partially supported by JSPS KAKENHI Grant Number 16H01756, 17H06103, 18H0333808, and JST CREST.

References

1. Chen, Y., Zhou, Y., Lu, Z., Sun, H., Yang, W.: Legal information retrieval by association rules. In: International Workshop on Juris-informatics (JURISIN 2018) (2018)
2. Joachims, T.: 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). https://doi.org/10.1145/775047.775067, http://doi.acm.org/10.1145/775047.775067

3. Juliano Rabelo, Mi-Young Kim, H.B., Goebel, R.: Legal information extraction and entailment for statute law and case law. In: International Workshop on Juris-informatics (JURISIN 2018) (2018)
4. Kano, Y., Kim, M.Y., Goebel, R., Satoh, K.: Overview of coliee 2017. In: Satoh, K., Kim, M.Y., Kano, Y., Goebel, R., Oliveira, T. (eds.) COLIEE 2017. 4th Competition on Legal Information Extraction and Entailment. EPiC Series in Computing, vol. 47, pp. 1–8. EasyChair (2017). https://doi.org/10.29007/fm8f, https://easychair.org/publications/paper/Fglr
5. Kim, M., Goebel, R.: Two-step cascaded textual entailment for legal bar exam question answering. In: Proceedings of the 16th edition of the International Conference on Articial Intelligence and Law, ICAIL 2017, London, United Kingdom, June 12-16, 2017. pp. 283–290 (2017). https://doi.org/10.1145/3086512.3086550, http://doi.acm.org/10.1145/3086512.3086550
6. Kim, M.Y., Goebel, R., Kano, Y., Satoh, K.: Coliee-2016: evaluation of the competition on legal information extraction and entailment. In: International Workshop on Juris-informatics (JURISIN 2016) (2016)
7. Kim, M.Y., Goebel, R., Satoh, K.: Coliee-2015: evaluation of legal question answering. In: Ninth International Workshop on Juris-informatics (JURISIN 2015) (2015)
8. Reina Hoshino, Ryosuke Taniguchi, N.K., Kano, Y.: Question answering system for legal bar examination using predicate argument structure. In: International Workshop on Juris-informatics (JURISIN 2018) (2018)
9. Ryosuke Taniguchi, R.H., Kano, Y.: Legal question answering system using framenet. In: International Workshop on Juris-informatics (JURISIN 2018) (2018)
10. Strohman, T., Metzler, D., Turtle, H., Croft, W.B.: Indri: A language model-based search engine for complex queries. In: Proceedings of the International Conference on Intelligent Analysis. pp. 2–6 (2005)
11. Vu Tran, S.T.N., Nguyen, M.L.: Jnlp group: Legal information retrieval with summary and logical structure analysis. In: International Workshop on Juris-informatics (JURISIN 2018) (2018)
12. Yoshioka, M.: Analysis of coliee information retrieval task data. In: Arai, S., Kojima, K., Mineshima, K., Bekki, D., Satoh, K., Ohta, Y. (eds.) New Frontiers in Artificial Intelligence. pp. 5–19. Springer International Publishing, Cham (2018). https://doi.org/10.1007/978-3-319-93794-6_1
13. Yoshioka, M., Song, Z.: Hukb at coliee2018 information retrieval task. In: International Workshop on Juris-informatics (JURISIN 2018) (2018)

JNLP Group: Legal Information Retrieval with Summary and Logical Structure Analysis

Vu Duc Tran, Son Truong Nguyen, and Minh Le Nguyen

School of Information Science,
Japan Advanced Institute of Science and Technology (JAIST),
Nomi, Ishikawa, Japan.
{vu.tran, nguyen.son, nguyenml}@jaist.ac.jp

Abstract. We present our methods for solving the legal case retrieval and statute law retrieval tasks in Competition on Legal Information Extraction/Entailment, 2018. We explore the benefits from analyzing legal documents' summaries and logical structures to tackle these legal information retrieval tasks.

Keywords: legal texts, deep learning, document representation, structure analysis, information retrieval

1 Introduction

Automatic legal document processing systems can speed up significantly the work of experts, which, otherwise, requires significant time and efforts. One crucial kind of such systems, automatic information retrieval whose systems, in place of experts, process over enormous amount of documents, for example, legal case reports and statute law documents, which are accumulated rapidly over time.

In this paper, we describe our method for tackling the legal case retrieval and statute law retrieval tasks in Competition on Legal Information Extraction/Entailment, 2018. These are the main points of our work:

- *Legal case retrieval task*
  - We developed an approach to obtain document representation in continuous vector space in which the summary properties of the document are embedded, which we call encoded summarization.
  - We extracted and compared lexical features on different parts of queries and candidates including: summary, fact, and paragraphs.
  - We found that combining the lexical features and the encoded summarization achieved better results than only each of them.
- *Statute law retrieval task*
  - We improved the system in [10] that allow to use the result of legal structure analysis as an additional information for measure the relevance between a query and an article.

- We found that sometimes that additional information can improve the performance of the task.

The remainder of this paper are as follows. Section 2 describes our proposed models, experiments and results for the legal case retrieval task. Section 3 describes the proposed models, experiments and results for the statute law retrieval task. Finally, Section 4 concludes our work.

2 Legal Case Retrieval Task

2.1 Task Overview

The legal case retrieval task involves reading a new case q, and extracting supporting cases c_1^*, c_2^*, ..., c_n^* for the decision of q from a given list of candidate cases. The candidate cases that support for the decision of a new case are called 'noticed cases'. The legal cases are sampled from a database of predominantly Federal Court of Canada case laws, provided by Compass Law.

When dealing with this task, we observed several obstacles. First, the candidate cases are 5.7K-token long in average. This poses the problem of understanding the reason of selecting the cases as supporting cases. We, then, chose another approach which is comparing the summaries of each query and its candidate cases. We, however, found that the summary of the query is not necessarily lexically similar to the summary of the candidate cases. Moreover, some candidate cases do not have summary at all. We would like to obtain the summary for each and every candidate cases, and furthermore, the summary should be comparable with the summary of the corresponding query. We come up with the idea of encoding the entire document into vector space embedding the properties of summarization, and we called it encoded summarization. Thus, we developed a combination method utilizing the lexical matching and encoded summarization.

2.2 Lexical Matching

We estimate the lexical matching between a query and a candidate case in different types of n-grams, skip-grams, longest common subsequence to measure various degrees of lexical similarity. We employ the following matching formulas from ROUGE evaluation[5]:

- ROUGE-1: Unigram matching. This measures word overlapping between a query and a candidate case.
- ROUGE-2: Bigram matching. Similar to ROUGE-1 with word replaced by two consecutive words as one token.
- ROUGE-L: Longest common subsequence. This measures the strictly ordered overlapping scattering over the texts.
- ROUGE-W: Weighted longest common subsequence. This favors matches with subsequences having less distance between words. While ROUGE-L can provide the information about strictly ordered overlapping scattering around, ROUGE-W can provide more precise overlapping, for example, consecutive matches.

– ROUGE-S: Skip-Bigram. This measures the co-occurrence of all pair of words in their sentence order.
– ROUGE-SU: Skipped bi-gram + uni-gram. This is a kind of combination of ROUGE-1 and ROUGE-S, the tokens are skip-bigrams and unigrams.

To have more precise comparison between a query and a candidate, we compute 4 matching options:

– Summary-Summary: matching of the query's summary with the candidate's summary. This matching represents the comparison of the highlights between the query and the candidate.
– Fact-Summary: matching of the query's fact with the candidate's summary. This matching represents the ratio of the candidate summary mentioning relevant fact.
– Summary-Paragraphs: matching of the query's summary with the candidate's paragraphs. This matching represents the ratio of the query's highlights mentioned in the candidate's detail.
– Fact-Paragraphs: matching of the query's fact with the candidate's paragraphs. This matching represents the ratio of the query's fact also occurred in the candidate's detail.

We compute precision, recall, and f-measure of the above matching operators and matching options as lexical matching features, where each of these metrics represents different aspects of the matching between a source and a target. For example, with the option of fact-paragraphs as source-target, the precision measures how much of the fact is mentioned in the paragraphs, the recall measures how much content of the paragraphs is mentioned in the fact, and the f-measure is a balance measurement of both precision and recall.

2.3 Encoded Summarization

Summary contains the highlights of a document, which is an important factor in relevance analysis. As observed from the dataset, the query's summary may not similar to the relevant cases' summaries. We would like to extend the summary of both the query and the candidates to include more points from fact/paragraphs.

In [11], the authors present a way of obtaining catchphrases of legal case documents via phrase scoring using deep neural networks. Catchphrases represent important legal points of a legal case document and are usually drafted by experts, thus play important roles for understanding the case. To generate catchphrases, they built a scoring model to estimate phrasal scores of a legal case document and used the phrasal scores to construct predicted catchphrases for a new legal case. The scoring model is trained given expert drafted catchphrases of the document and optimized with the objective that catchphrases are expected to have higher scores than other contents in the document. In other words, we can train such a scoring model given training documents with indicated important contents which are document summaries in our task.

Inspired from the work's learning process, we propose to obtain document embedding directed by document summary. We estimate the phrasal scores for each document given its summary (extracted using indicators '*Summary:*' and '*Present:*') and paragraphs. We then train the scoring model with the objective that summary contents are expected to have higher scores than paragraph contents. After obtaining the trained model, instead of following the second phase to select catchphrases, we generate encoded summary in continuous vector space by composing document vectors from latent representations of the model. The encoded summary represents the summarization nature of the document.

Phrase Scoring Model We describe the phrase scoring model of [11] for this task.

Given a document, we denote $w_j^{s_i}$ as word j^{th} of sentence i^{th}. The features of word w_j of a sentence are captured using convolutional neural layer [2–4, 9] as follows:

$$\mathbf{f_{w_j}} = ReLU\left(\mathbf{W}^c \begin{bmatrix} \mathbf{v}(w_{j-l}) \\ ... \\ \mathbf{v}(w_j) \\ ... \\ \mathbf{v}(w_{j+l}) \end{bmatrix}\right) \tag{1}$$

where, $\mathbf{v}(\cdot) : \mapsto \mathbb{R}^d$: word embedding vector lookup map, l: corresponding to the window size containing $2l+1$ words, $[\cdot] \in \mathbb{R}^{dl}$: concatenated embedding vector, $\mathbf{W}^c \in \mathbb{R}^{c \times dl}$: convolution kernel matrix with c filters, $\mathbf{f}_{w_j} \in \mathbb{R}^c$: word feature vector, $ReLU$: rectified linear unit activation.

Sentence features $\mathbf{f}_{s_i}$ are, then, captured by applying max pooling over the whole sentence.

$$\mathbf{f}_{s_i} = \text{max-pooling}_j(\mathbf{f}_{w_j^{s_i}}) \tag{2}$$

where max-pooling are operated over each dimension of vectors $\mathbf{f}_{w_j^{s_i}}$.

With the same max-pooling operation as above, we compute document features as:

$$\mathbf{f}_d = \text{max-pooling}_i(\mathbf{f}_{s_i}) \tag{3}$$

Finally, we apply a multilayer perceptron (MLP) with one hidden and one output layer

$$MLP(\mathbf{x}) = \text{sigmoid}(\mathbf{W}_2 \cdot \tanh(\mathbf{W}_1 \cdot \mathbf{x} + \mathbf{b}_1) + \mathbf{b}_2) \tag{4}$$

to compute the score of each word $w_j^{s_i}$ as

$$P\left(w_j^{s_i}, s_i, d\right) = MLP\left(\begin{bmatrix} \mathbf{f}_{w_j^{s_i}} \\ \mathbf{f}_{s_i} \\ \mathbf{f}_d \end{bmatrix}\right) \tag{5}$$

where the hidden layer computes the word representative features respecting to its local use (surrounding neighbors), its enclosing sentence, and its document. The word representative features are fed to the output layer to compute word score (ranging from 0.0 to 1.0). The model is trained following the optimization process in [11] with summary acting as gold catchphrases and paragraphs acting as document sentences.

Document Vector Composition We present our method of composing document vectors from the phrase scoring model.

First, given a document, we obtain its the phrase scores and its internal representations: phrase level, sentence level and document level encodings.

Then, we compose the document vector as:

$$\mathbf{g}(d) = \frac{\sum_{i,j} P\left(w_j^{s_i}, s_i, d\right) \times \left[\mathbf{f}_d; \mathbf{f}_{s_i}; \mathbf{f}_{w_j^{s_i}}\right]}{\sum_{i,j} P\left(w_j^{s_i}, s_i, d\right)} \tag{6}$$

This composition resembles summarization where we weight the document internal representations by its summary. Thus, we call this composition encoded summarization.

Query-Candidate Relevance Vector Each dimension in one document vector is obtained from composition of the same kernel representing one feature. By comparing each dimension independently, we can estimate the compatibility of a query and a candidate over the dimension. Thus, we compute the query-candidate relevance vector as the element-wise product of query vector and candidate vector.

First, we obtain query vector $\mathbf{g}(q)$ and candidate vector $\mathbf{g}(c)$ using the document vector composition in Equation 6. Then, we compute the relevance vector of query q and candidate c by the following element-wise product.

$$\mathbf{h}(q, c) = \mathbf{g}(q) \odot \mathbf{g}(c) \tag{7}$$

2.4 Learning to Rank Candidates

We formulate the task as bipartite ranking problem and devise the learning to ranking method to solve it using lexical matching features from Sections 2.2 and query-candidate relevance features from Section 2.3.

We utilize pair-wise ranking strategy: pairing each noticed case with an irrelevant case from the candidate list.

We adopt Linear-SVM as the learning algorithm for solving the optimization problem.

After obtaining the scored candidates as a ranked list, we proceed to select top candidates as the predicted noticed cases. We use two selection strategies: 1) based on hard top k, 2) based on the score range from top 1 candidate and is bound relatively by score deviation.

– Top k: selecting top k highest scored candidates.
– Range r: selecting all cases whose scores in the interval:

$$Range(r) = (top\_1\_score - r \times mean\_score\_standard\_deviation, top\_1\_score] \tag{8}$$

where $mean\_score\_standard\_deviation$ is the mean of score standard deviations of the supporting cases of the training queries and is calculated in Equation 9. By applying a fixed score range over the ranked candidate cases, we only select candidate cases whose score differences among them do not exceed the range which is not achievable with the previous strategy.

$$mean\_score\_standard\_deviation = \frac{1}{|q|} \sum_{q} \sqrt{Var[c^*]} \tag{9}$$

2.5 Experiments

We evaluated our approach by performing leave-one-out validation. From the organizer provided 285 training queries, we used each and every one query as validation data and the other 284 queries as training data, which creates 285 folds for leave-one-out validation.

Table 1. Phrase scoring model parameters.

Parameter	**Description**
Embeddings (vector size d)	GloVe [8] $d = 300$
CNN filters c	300
CNN window size $2l + 1$	5
MLP hidden size	300
Optimizer	Adam[1]
Learning rate	0.0001
Gradient clipping max norm	5.0
Loss coefficients $(a_1, a_2, b_1, b_2, b_3, b_4)$	$(1.0, 1.0, 0.5, 0.1, 0.01, 0.02)$
Size of negative set $\|\{d'\}\|$	2

For the phrasal scoring model, we copied the model settings from [11] as shown in Table 1. For word embeddings, we use the pre-trained GloVe[1] which was trained on data from "Common Crawl"[2], an open repository of web crawl data. For computation of ROUGE scores, we used publicly available ROUGE

[1] `https://nlp.stanford.edu/projects/glove/` Common Crawl (840B tokens, 2.2M vocab, cased, 300d vectors)
[2] `http://commoncrawl.org/`

implementation[3] with Poster stemmer and stopword removal enabled[4]. We used the default word and sentence tokenization from NLTK[6][5].

We reported our system's validation results (Table 2) with the following metrics:

- MRR: Mean reciprocal rank.
- MAP: Mean average precision.
- P-macro, R-macro, F-macro: Precision, Recall, F-measure whose values are averaged by query. This is straightforward as we average the results of all folds in the leave-one-out validation.
- P-micro, R-micro, F1-micro: Precision, Recall, F1 whose values are computed by putting the outputs of all test queries in the leave-one-out validation into one for micro-based measurement.

The validation results (Table 2) show that performance is improved when using encoded summarization over lexical matching and even further improved when combining both of them. The combination improve 0.185 F1-macro and 0.182 F1-micro compared to lexical matching and 0.105 F1-macro and 0.091 F1-micro compared to encoded summarization. While the improvement from lexical matching to encoded summarization suggests the importance of summarization in supporting case retrieval, the further improvement of the combination hints the existence of latent features not captured by lexical approach.

Table 2. Leave-one-out validation results on provided 285 training queries. Precision (P), recall (R) and f-measure (F1) are calculated with candidate selection strategy 1: top $k = 10$ (the average number of supporting cases in training data).

Model	**MRR**	**MAP**	**P macro**	**R macro**	**F1 macro**	**P micro**	**R micro**	**F1 micro**
Lexical Matching	0.764	0.530	0.420	0.520	0.398	0.420	0.426	0.423
Encoded Summarization	0.874	0.659	0.510	0.584	0.478	0.510	0.517	0.514
Lexical Matching +Encoded Summarization	0.966	0.849	0.601	0.761	0.583	0.601	0.609	0.605

The submission has two folds corresponding to the two candidate selection strategies. We selected $k = 10$ which is the average number of noticed cases from the training data and $r = 2.5$ which results in the best F1-macro in the validation process (Fig. 1). The submission results are shown in Table 3.

[3] `https://github.com/andersjo/pyrouge`
[4] rouge parameters: -c 95 -2 -1 -U -r 1000 -n 2 -w 1.2 -a -d -m -s
[5] `https://www.nltk.org/` version 3.3.0

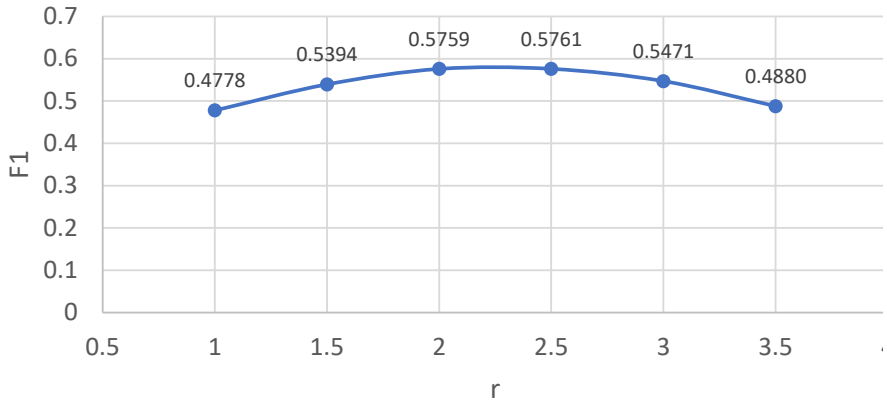


Fig. 1. Leave-one-out validation results of candidate selection strategy 2 on provided 285 training queries. We evaluated each $r \in \{1, 1.5, ..., 3.5\}$ based on the competition's primary measure F1.

Table 3. Submission results on test data. We submitted the outputs of the combination model "Lexical Matching + Encoded Summarization". The submission has two folds corresponding to the two candidate selection strategies. We selected $k = 10$ which is the average number of noticed cases from the training data and $r = 2.5$ which results in the best F1-macro in the validation process (Fig. 1).

Submission	P-micro	R-micro	F1-micro
$k = 10$	0.675	0.634	0.655
$r = 2.5$	0.546	0.655	0.596

3 Statute Law Retrieval Task

3.1 Task Overview

A query in the Statue Law Retrieval task is usually represented as a statement that a user want to check whether or not it is correct. The task will try find articles in Japanese Civil Code that are relevant to the query to answer it. This is also a challenging task. Firstly, in many cases, articles describe the general law provision, but the query describes a concrete situation. Therefore, the lexical matching methods may fail to find relevant articles. Secondly, in some cases, although the query are quite similar to the article by lexical matching method, they are not relevant. Table 4 shows an example in which the article *A961* shared much more words than article *A974*, however, it is not relevant to the given query.

Table 4. An example that lexical matching methods fail to find relevant articles

QUERY: A minor who has attained 15 years of age may be a **witness to a will**	
ARTICLE	LABEL
* A961: Any person who has attained 15 years of age **may make a will**.	NOT RELEVANT
* A974: The following persons **may not be a witness or observer to a will** ...	RELEVANT

We found that in these cases, the effectuation parts of articles (the **bold** parts in the example) are more important than requisite parts in measuring the relevance between the query and articles. Our proposed system also focuses this issue by using the effectuation parts as additional information to compute the relevance.

3.2 Statute Law Retrieval based on Requisite-Effectuation Analysis

We use the conventional Information Retrieval approach to solve this task with the usage of some additional information which obtained from analyzing components of legal sentences. Requisite and effectuation parts are main components in legal sentences and we hypothesize that these parts may affect to the performance of IR task. The system is based on [10] with several steps:

Processing : We conduct some basic processing techniques such as stemming, and removing stop-words.

Indexing : We using uni-gram, bi-gram, three-gram word indexing model with the TF-IDF term weighting technique to represent articles as vectors. We use two different scopes to index articles:

- Indexing based on the whole content of each article.
- Indexing based on effectuation parts: We use only effectuation parts of each article. Effectuation parts of articles are obtained by analyzing JPC-RRE corpus [7].

Scoring models and Relevant Analysis: We compute cosine similarity scores between vector representations of the given query and each article as follows:

- $s_1(q, a) = cosine(q, a)$: computes the similarly between the query q and the whole content of the article a.
- $s_2(q, a) = cosine(q, a_{\mathbf{E}})$: computes the similarly between the query q and effectuation parts of the article a.
- $\mathbf{similarity}(q, a) = s_1(q, a) + \alpha \times s_2(q, a)$: This is the combination of two above scores. The value of α indicates how much the contribution of effectuation parts. If α is 0, this is the same with original method which described in [10].

After similarity scores are computed, a number of top-k relevant articles are selected as relevant articles.

3.3 Experiments and Results

Parameters tuning We use the training set for tuning parameters and model selection. Firstly, we tune the indexing model and top-k, the left figure in Fig. 2 shows that the bi-gram word indexing model and $top - k = 2$ produce the best performance (in F_2 measure). Secondly, we tune the parameter α to find that how much how much the information of effectuation parts will be add into to original score to obtain the best performance. The right figure in Fig. 2 shows that the best combination of two similarity score is obtained when $\alpha = 0.4$.

Results Table 5 shows the evaluation result on the official test set which includes 69 queries. Although the effectuation parts show positive effects in the training set, it reduces the performance on the test set. However, it improves the **MAP** scores.

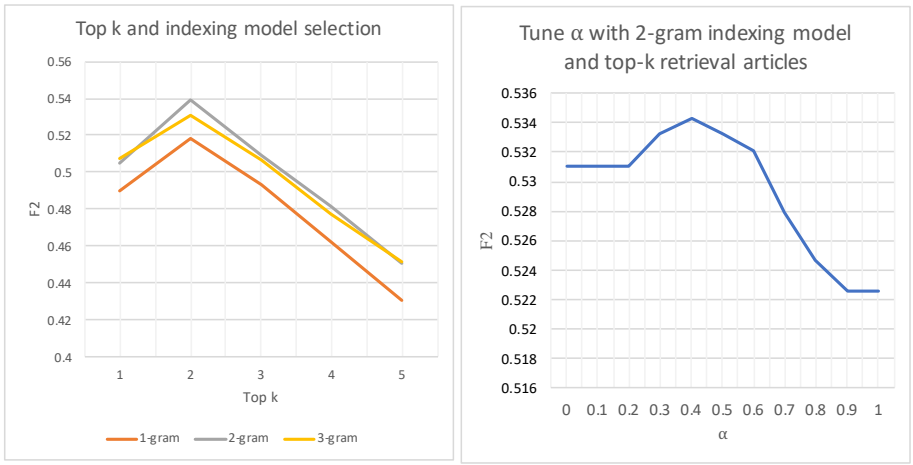


Fig. 2. Hyper parameter tuning including (1) tuning top-k and indexing model (left figure) and (2) tuning α for scoring model. The best parameter set are $\{top-k=2, indexingmodel: 2-gram, \alpha=0.4\}$

Table 5. Performance on the test set. Different values of α indicate the contributions of effectuation parts in computing similarity scores.

Submission	**Precision**	**Recall**	**F2**
$\alpha = 0.0$	0.413	0.6404	0.5769
$\alpha = 0.4$	0.4058	0.6292	0.5668

Submission	**R@2**	**R@5**	**R@10**	**R@30**	**R@100**	**MAP-L**
$\alpha = 0.0$	0.6404	0.764	0.8202	0.9213	0.9551	0.7398
$\alpha = 0.4$	0.6292	0.7528	0.809	0.9101	0.9775	0.7296

4 Conclusion

We have presented our methods for legal case retrieval and statute law retrieval tasks in Competition on Legal Information Extraction/Entailment, 2018.

For the legal case retrieval task, we developed an approach of combining lexical features and latent features embedding summary properties which we call encoded summarization. The validation results show that performance is improved when using encoded summarization over lexical matching and even further improved when combining both of them. While the improvement from lexical matching to encoded summarization suggests the importance of summarization in supporting case retrieval, the further improvement of the combination hints the existence of latent features not captured by lexical approach.

For the law statute retrieval task, we incorporated the structure of legal sentences in computing similarity between queries and articles. However it only show the improvement on the training set. In future, we will investigate different types of additional information such as subjects, verbs and objects of legal sentences because each of type may possess different contributions for the relevant analysis in this task.

Acknowledgements

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

References

1. Duchi, J., Hazan, E., Singer, Y.: Adaptive subgradient methods for online learning and stochastic optimization. Journal of Machine Learning Research 12(Jul), 2121–2159 (2011)
2. Johnson, R., Zhang, T.: Effective use of word order for text categorization with convolutional neural networks. In: Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. pp. 103–112. Association for Computational Linguistics, Denver, Colorado (May–June 2015), http://www.aclweb.org/anthology/N15-1011
3. Kalchbrenner, N., Grefenstette, E., Blunsom, P.: A convolutional neural network for modelling sentences. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 655–665. Association for Computational Linguistics, Baltimore, Maryland (June 2014), http://www.aclweb.org/anthology/P14-1062
4. Kim, Y.: Convolutional neural networks for sentence classification. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). pp. 1746–1751. Association for Computational Linguistics, Doha, Qatar (October 2014), http://www.aclweb.org/anthology/D14-1181
5. Lin, C.Y.: Rouge: A package for automatic evaluation of summaries. Text Summarization Branches Out (2004)
6. Loper, E., Bird, S.: Nltk: The natural language toolkit. In: Proceedings of the ACL-02 Workshop on Effective Tools and Methodologies for Teaching Natural Language

Processing and Computational Linguistics - Volume 1. pp. 63–70. ETMTNLP '02, Association for Computational Linguistics, Stroudsburg, PA, USA (2002), https://doi.org/10.3115/1118108.1118117
7. Nguyen, T.S., Nguyen, L.M., Tojo, S., Satoh, K., Shimazu, A.: Recurrent neural network-based models for recognizing requisite and effectuation parts in legal texts. Artificial Intelligence and Law 26(2), 169–199 (Jun 2018), https://doi.org/10.1007/s10506-018-9225-1
8. Pennington, J., Socher, R., Manning, C.D.: Glove: Global vectors for word representation. In: Empirical Methods in Natural Language Processing (EMNLP). pp. 1532–1543 (2014), http://www.aclweb.org/anthology/D14-1162
9. Severyn, A., Moschitti, A.: Learning to rank short text pairs with convolutional deep neural networks. In: Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 373–382. ACM (2015)
10. Son, N.T., Anh, P.V., Minh, N.L.: Recognizing entailments in legal texts using sentence encoding-based and decomposable attention models. In: Satoh, K., Kim, M.Y., Kano, Y., Goebel, R., Oliveira, T. (eds.) COLIEE 2017. 4th Competition on Legal Information Extraction and Entailment. EPiC Series in Computing, vol. 47, pp. 31–42. EasyChair (2017), https://easychair.org/publications/paper/wHFc
11. Tran, V.D., Nguyen, M.L., Satoh, K.: Automatic catchphrase extraction from legal case documents via scoring using deep neural networks. In: Workshop on MIning and REasoning with Legal texts (2018), https://www.researchgate.net/publication/327645790_Automatic_Catchphrase_Extraction_from_Legal_Case_Documents_via_Scoring_using_Deep_Neural_Networks

Legal Information Extraction and Entailment for Statute Law and Case Law

Juliano Rabelo[1], Mi-Young Kim[1,2], Housam Babiker[1,3], Randy Goebel[1,3], and Nawshad Farruque[1,3]

[1]Alberta Machine Intelligence Institute, University of Alberta, Edmonton, AB, Canada
[2]Department of Science, Augustana Faculty, University of Alberta, Camrose, AB, Canada
[3]Department of Computing Science, University of Alberta, Edmonton, AB, Canada
{rabelo,miyoung2,khalifab,rgoebel,nawshad}@ualberta.ca

Abstract. The problem of information overload in the legal domain is increasing every day. In the Competition on Legal Information Extraction and Entailment (COLIEE), we tackle four challenges which are intended to encourage the development of systems and methods to alleviate that pressure: case law retrieval (Task 1) and entailment (Task 2), and statute law retrieval (Task 3) and entailment (Task 4). For Tasks 1 and 2, we propose a pairwise paragraph similarity based approach, which ranked 4th place in Task 1 and 1st place in Task 2 among all teams participating in COLIEE 2018. For the statute law textual entailment challenge, we use a textual segmentation of legal data based on the characteristics of statute law, then confirm textual entailment based on negation/antonym relations. We have evaluated our system using the data from the COLIEE 2018. The statute law competition in COLIEE focuses on the legal information processing required to answer yes/no questions from legal bar exams, and it consists of two phases: legal ad-hoc information retrieval (Task 3), and textual entailment (Task 4). We focus on Task 4, and our experimental evaluation demonstrates the effectiveness of the legal segmentation and negation/antonym detection. Our results show that our method was ranked first in the Task 4 of the COLIEE 2018 competition.

Keywords: Legal textual retrieval · Legal textual entailment · Document similarity · Binary classification · Imbalanced datasets

1 Introduction

Every day, large volumes of legal data are produced by law firms, law courts, independent attorneys, legislators, and many other sources. In that scenario, the amount of information generated becomes manually intractable, and calls for the development of tools which automatically or semi-automatically aid legal professionals to handle the information overflow. In COLIEE, we face four facets of that challenge: case law retrieval, case law entailment, statute law retrieval

and statute law entailment. Here we provide the details of our approaches to those problems, evaluate the results achieved and comment on future work to further improve our models.

For the case law retrieval (Task 1), our approach relies on measuring the pairwise similarity between paragraphs of two case laws, generating feature vectors based on those similarities, and then using a classifier to determine whether those cases should be noticed or not. We compare that strategy with a baseline algorithm with uses the full text of the cases on the similarity calculation to confirm the hypothesis that considering individual paragraph similarity provides a more meaningful way of comparing case laws.

A similarity based strategy is also applied for the entailment task (Task 2). In that case, we augment the model with word embeddings and use the available decision and summary information, in addition to the case law contents, for the similarity calculation. For both tasks 1 and 2, we discuss the problem of class imbalance and propose/evaluate strategies to cope with that problem.

In the statute law information extraction, our approach to Yes/No legal question answering requires two steps. First, an information retrieval system must retrieve relevant documents for a legal question (Task 3). Second, a textual entailment system compares the semantic connections between question and relevant documents, and determine whether an entailment relation holds (Task 4).

Here, we focus on Task 4. First, we segment the statute law based on legal condition/conclusion/exception, and then detect negation/antonym relation for textual entailment. The detailed method will be explained in Section 4.

2 Case Law Retrieval

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

In the following subsections, we will present an overview of the competition dataset, discuss details of our method and analyze the results it achieved.

2.1 Dataset Analysis

The dataset provided for this task has 285 query cases, each consisting of a summary file (i.e., a short, summarized version of the case) and a fact file containing a list of the case paragraphs. For each query case, a set of 200 candidate cases is given, from which must be identified a subset of the noticed cases. The candidate cases are given in raw text, i.e., they are a full text dump of the case laws descriptions, containing not only the paragraph list, but also parties, judges, dates, key concepts, etc. Table 1 summarizes the dataset properties.

Table 1. Summary for the Case Law Retrieval Task dataset.

Property	Value
Number of query cases	285
Candidate cases per query	200
Total number of candidate cases	57,000
Total number of true noticed cases	2,813 (4.93%)
Average number of noticed cases per query	9.87
Standard deviation of noticed cases counts	8.8143

2.2 Our Method

Before describing the details of our approach, it is important to clarify one aspect about our task and how one could tackle it. As noted in section 2, we are required to identify which candidate cases should be noticed in respect to each given query case. But what makes a case to be noticed in respect to another case? Noticing a case is something a lawyer does when he/she understands that case is somehow relevant to the case being analyzed. "Somehow relevant" is to some degree subjective: a court is obliged to consider cases in which the same issue has been decided in a court whose decisions are binding. Similarly, a court is obliged to consider its own precedent. But a lawyer may also find a case is relevant and should be noticed because it is on the same legal subject, because it addresses similar facts, because it addresses similar parties, because it requires a similar type of analysis, or it is for some other reason usefully analogous. Our method to identify the noticed cases is built on an important assumption: we consider a case should be noticed if it is similar to the query case. Similarity is somewhat vague, but seems to be a generic concept that captures most variations of the valid reasons for noticing a case. Besides, there are several well-known existing document similarity measures [1] that we can adopt for this task.

The Baseline Approach The baseline algorithm implemented for the case law retrieval task is a simple document similarity calculator: if the similarity measured between the query case and the candidate case is above a threshold, the documents are considered similar (i.e., they should be noticed). We perform some preprocessing on the documents before measuring the similarity: removal of punctuation, digits and stop words, case normalization and stemming. One of the implemented strategies (see subsection 2.2) used a pre-trained word embedding model, for which we did not apply stemming.

In addition to these well known preprocessing techniques, we also implemented an algorithm to join hyphenated words into a single word, since the input documents contain hyphenated words (e.g., "attor-ney," "deci-sion"). The algorithm only joins two tokens separated by an hyphen if the resulting token is a valid English word, which is determined by a dictionary lookup. That is an attempt to avoid joining two words which should really be separated by a dash (e.g., "ex-president," "vice-consul," "commander-in-chief"). The dictionary for the word lookup was built by parsing an English-German dictionary [15] and

parsing it to keep only the English words. The outcome of that process was an English word list of $\sim 170,000$ terms.

In the baseline approach, the similarity between the base case and a candidate noticed case is calculated as the cosine of the vectors formed by a bag of words representation of each document [2]. The results achieved by this simple approach for different threshold values are shown on Table 2.

Table 2. Baseline results.

Threshold / Measure	0.20	0.25	0.30	0.35	0.40	0.45	0.50
Recall	0.8115	0.6352	0.4827	0.3668	0.2609	0.1809	0.1038
Precision	0.0897	0.1235	0.1743	0.2459	0.3387	0.4480	0.5407
F-measure	0.1616	0.2068	0.2561	0.2944	0.2947	0.2577	0.1741

Those results show the best F-measure scores were 0.2944 and 0.2947, achieved for thresholds of 0.35 and 0.40, respectively. The results are not great but they provide a basis for comparison with other, more elaborate methods.

Pair-wise Paragraph Comparison Calculating the similarities using the cases full text is intuitive, but in fact there are studies showing that the main concepts of case laws are usually encoded in a few paragraphs [3]. To leverage that idea, we implemented a similarity based approach which compares the query case and each candidate case paragraph-wise; thus we needed to implement a parser to read the query cases and the candidate cases, and use only the paragraph lists from both for the comparison.

That comparison step would now generate not a similarity score, but a similarity matrix. We could then use an aggregation function (such as average) on it, which would enable us to set a threshold and make a decision on whether the cases are similar, but would also sacrifice the more detailed information, making this approach similar to the baseline approach presented above.

Another option which would leverage the paragraph-wise similarities would be to use that as an input to a supervised learning classifier. However, the similarity matrices generated have a variable number of rows and columns (since their sizes are defined by how many paragraphs the query and the candidate cases have). So we needed to transform each similarity matrix into a regularized input to the classifier; this was achieved by creating a histogram over the similarity matrices. We define the bounds of the similarity bins and count how many cells in the matrix fall under each bin, then we normalize the histogram by the number of cells in the matrix. This process produces a fixed size input vector which can then be fed to a supervised learning classifier.

To leverage other studies [3], we also measure the paragraphs similarity considering only "legal terms"[1], i.e., we use a list of law-related terms and preprocess the paragraphs retaining only relevant terms. The similarities scores were collected and processed in the same way as the "regular" paragraph similarities, thus augmenting the feature vectors which are then fed to the learning classifier.

One problem which emerged was the significant class imbalance in the competition dataset (see Section 2.1): there are only 4.93% noticed cases in the dataset. To cope with that problem, we applied under-sampling over the "false" class (i.e., the candidates which are not noticed cases). We did not want to shrink the training dataset to only as many positive samples as there are available, so we kept the negative dataset size five times bigger than the positive dataset. We also attempted to oversample the positive dataset, first by replicating the positive samples and then by synthesizing samples using SMOTE [4]. None of those approaches proved to be effective for this task. We also tried to use a meta cost sensitive classifier [5], but it only made sense when we used a fully balanced dataset for training, so we dropped that idea in favor of the use of a bigger training dataset.

For the classification itself, we tried a few different classifiers with none or little tweaking in the default parameters: Random Forest [6], Gradient Boosting [7] and GCForest [8]. Given the gain in performance was relevant (see Table 3), we chose to run the rest of the experiments using only the Random Forest classifier.

Table 3. Results for the Pairwise Paragraph Comparison Approach.

Algorithm / Measure	Precision	Recall	F-measure
Random Forest	0.4200	0.3194	0.3629
Gradient Boosting	0.1422	0.3541	0.2029
GC Forest	0.3684	0.2430	0.2928

Pair-wise Paragraph Comparison Augmented with Word Embeddings
One problem of the similarity based approach is there might be paragraphs which express similar concepts using very different phrasings. Since words with similar concepts should have similar vector representations in word embeddings [9], its application could help overcome that problem. The framework used in our experiments was Google's Word2vec [9] and the similarity was calculated using the Word Mover's Distance [10]: basically the smallest distance all embeddings from one document (in our case, paragraph) need to "travel" to reach the other document's embeddings.

[1] the legal terms list was created by downloading a few online available law dictionaries and parsing their contents

In our experiments, we used the word embeddings model pre-trained on a general purpose set of documents [12], we figured we could have better results by training our own model using case law data, thus improving the representation for case law specific terms. So we downloaded and parsed Case Law Reports [11], but that was not enough to out perform the general purpose model.

Post processing By analyzing the available dataset (Table 1), we see there are on average 10 noticed cases for each query case; so we assumed a range for minimum and maximum number of noticed cases for each query case. That could help, especially in decreasing false positives if there are cases for which the algorithm identifies too many noticed cases. But for this task, the post processing did not help as much as it did for Task 2. Our assumption is that the much higher standard deviation seen in the Task 1 dataset impaired the results.

2.3 Results

Below we show the results achieved by all variations of the method on the training dataset, and the variations selected for submission on the COLIEE test dataset.

On the Training Dataset To evaluate the results on the training dataset, we set apart 29 cases from the original 285 cases (see Table 1). Those cases were not used on the development of our method (training classifiers, manual inspection, etc). The results achieved on that validation dataset are shown on Table 4.

Table 4. Results for the Case Law Retrieval Task (validation dataset).

Algorithm	Precision	Recall	F-Measure
Baseline Approach	0.2609	0.3387	0.2947
Pair-wise Paragraph Comparison (PPC)	0.4200	0.3194	0.3629
PPC with word embeddings	0.4198	0.3090	0.3560
PPC with SMOTE	0.3660	0.3888	0.3771
PPC with Post Processing	0.3087	0.3784	0.3400

From those experiments, we selected to submit to the official COLIEE evaluation the results of PPC, PPC with SMOTE and PPC with Post Processing. PPC with word embeddings did not provide improvements on the F-measure, in comparison with PPC, and was computationally quite expensive.

On the COLIEE Test Dataset The results achieved by the three selected methods are presented on Table 5.

Table 5. Results for the Case Law Retrieval Task (real COLIEE test dataset).

Method	Precision	Recall	F-Measure
Pair-wise Paragraph Comparison (PPC)	0.3724	0.3227	0.3458
PPC with SMOTE	0.3538	0.3926	0.3722
PPC with Post Processing	0.3484	0.4038	0.3740

2.4 Future Work

We plan to explore other alternatives for this task. The result analysis showed the similarity based features do provide relevant information for the classifiers but were not great discriminants for the problem. To overcome that problem, we might extract other features. One alternative is to extract LWIC features [13]. A more immediate option is the use of the summary data (we only used the case law main contents in our experiments).

We also intend to perform other experiments with word embeddings. Although the results achieved in our experiments were not so promising, we believe the idea may be refined. The first thing would be to use specific-domain data to train the model. We do have some data available, which has been downloaded from [11], and we could augment it with the 57,000 candidate case laws provided as this task's training dataset. Another slightly different option is to re-train a general purpose word embedding model with our data. We believe those alternatives would improve the vector representations of the model for the problem at hand. We could also try coarser grained embeddings (sentence or document level embeddings [20], as opposed to the current word-based implementation). However, we would need significantly more data to train those embeddings, which makes it a poor fit for the COLIEE scenario, where relatively little data is available. A more viable alternative would be the application of different frameworks, such as GloVe [16] or Fasttext [17], which may capture better vector representations in our scenario, or trying a different learning model for the embeddings. We used the default CBOW model, but an option worth trying would be to use skip-gram (which usually works better than CBOW for smaller datasets [18]).

One of the major problems which impacted our results was the severe class imbalance. For the training to be performed reasonably, we needed to balance the training dataset with undersampling/oversampling. Still, the classifier is applied to a real test dataset which presents the original imbalance. Overcoming that situation requires modeling the problem in a different way. One option would be modeling it as outlier detection. We could train a one-class classifier (e.g., One Class SVM [19]) with the false samples (the more abundant class). Samples classified as not belonging to that class would be considered true noticed cases. Another alternative would be applying clustering to the training dataset, possibly allowing multiple clusters for each class to capture diversity in the training data. New samples would be classified with the class of the nearest clusters.

A more sophisticated alternative would be the extraction of the ratio decidendi [21] for each case. If we can successfully implement that and an effective similarity measure between ratio decidendis, it would be simple to establish some

sort of threshold to determine which cases should be noticed. However, the single previous description on automatically extracting the ratio decidendi [21] has been tested on only one case law; the technique may not be adequate to an open world scenario. Extracting a ratio decidendi means fully understanding a case law, and implementing a general approach to that problem is not something already solved in the literature. But in general, the development of methods for fully understanding case law is precisely the ultimate goal of Task 1 on COLIEE.

3 Case Law Entailment

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

In the following subsections, we will present an overview of the provided dataset, discuss details of our method and analyze the results it achieved.

3.1 Dataset Analysis

The dataset provided for this task has 181 base cases, each consisting of a summary file, a fact file (a list of the case paragraphs), and a decision file. For each base case, it is provided a respective related case represented by a list of paragraphs, from which the paragraphs that entail the base case decision must be identified. Table 6 summarizes the dataset properties.

Table 6. Summary for the Case Law Entailment Task dataset.

Property	Value
Number of base cases	181
Total number of paragraphs in the related cases	8,794
Total number of true entailment paragraphs	239 (2.71%)
Average number of entailment paragraphs per base case	1.32
Standard deviation of entailment paragraphs counts	0.7106

3.2 Our Method

This task is even more challenging than Task 1. Solving it would ideally require in-depth domain and common sense knowledge, plus a reasoning capability in order to mimic the whole process undertaken by a human annotator who analyses the data to determine which paragraphs from one case entail a decision of

a different case. Assuming we do not currently have tools to implement such a complex model, we decided to try simpler approaches which do not attempt to reason and come to a conclusion. Our method tries to extract relevant features which (hopefully) correlate with the entailment relationship, and then train a classifier which is capable to learn that relationship. We also face the same problems of data scarcity and class imbalance we faced in Task 1, but those problems are more extreme for Task 2. Table 6 shows we have less data for this task and class imbalance is more severe. We experimented three oversampling techniques to cope with the imbalance problem: besides simple replication and SMOTE (see section 2.2), we also implemented a new instance synthesis algorithm: for every instance p in the positive dataset, compute a new instance of p as $\mu_p + p$.

Preprocessing The preprocessing used here was a little more elaborate that it was for Task 1: in addition to using the simple techniques used for Task 1 (case normalization, stop words, punctuation, accents and digit removal, and hyphenated words joining), here we also implemented a language detection algorithm. This was necessary because the dataset contains many paragraphs written in French[2]. Since our approach is based on the similarity between documents, and the input cases are in English only, we decided to not consider the French candidates, although there are 6 expected answers which are identified by our algorithm as French written paragraphs for Task 2. This was considered an acceptable limitation of our method, since those cases represent only 2.5% of the total positive samples. Moreover, the task is already hard enough, so we decided not to tackle cross language entailment for now. After the removal of the French paragraphs, we could use the English instances to train our model[3].

Looking for Features Our first attempt relied solely on searching for words which could indicate a paragraph was written in a way the judge would be expressing his/her decision. This would not take into consideration any relationship with the base case, but instead would only look for "decision-like" paragraphs. It was not expected that this would be the final method for Task 2, but only a means of verifying the correlation between some words and the task goal. In fact, only by looking up a few words such as first person pronouns ("I," "my" etc.), we could identify around 25% of the entailment paragraphs (a good recall for such a simple approach). However, the precision was around 2%, meaning this method alone was far from ideal (our f-measure would be 0.0370). However, it showed there might be words correlated with the entailment paragraphs.

Word Clouds A slightly more sophisticated approach in that direction was based on the generation of word clouds for the true and false samples. We divided

[2] our algorithm could identify 145 instances of French paragraphs in the dataset

[3] the official COLIEE test dataset did not present any cases of French paragraphs. Still, removing those paragraphs was important because we used the training dataset, which does contain French paragraphs, for training our classifiers.

all the candidate paragraphs according to their label (true entailment or not) and generated an aggregate word cloud (or bag of words) for each class. Then, we took the first n words from each cloud, removed duplicates, and generated a dictionary. Each paragraph in the related cases were then encoded as one-hot vectors according to that dictionary; the resulting vectors were fed to a classifier.

Since the resulting vectors were very sparse, we augmented this approach with word embeddings: for a word w in the paragraph which was not in the dictionary, we looked up the most similar word on the embeddings (i.e., the closest concept) and used the similarity score as the feature for w in our feature vector. Therefore the feature vectors were not one-hot encoded, being now in the $[0 - 1]$ range. Results on the evaluation dataset are detailed on section 3.3.

The Similarity Based Approach Since those approaches did not provide acceptable results, our next try was based on the similarity method presented on section 2. We measured the similarity between each paragraph in the noticed case and the base case, generating a feature vector comprised of the measures of similarity between the paragraph and (1) the decision of the base case, (2) the base case summary and (3) the base case paragraphs (actually a histogram of the similarities between each noticed case paragraph and all paragraphs from the base case). In the end, we had one feature vector representing each "candidate" paragraph (i.e., a paragraph in the noticed case, which in this task is a candidate for entailing the decision on the base case). That approach provided much better results. We ran experiments using Random Forest [6] and Gradient Boosting [7] classifiers. See section 3.3 below for details.

LIWC Features In an attempt to further improve those results, we augmented the feature vectors with LIWC features [13]. Those are features usually applied in medical (psychiatric/psychological) applications [14] and include long words count, pronouns count (aligned with the idea in our first approach), sentiment analysis related features and many others. Those features did provide relevant improvements on the model performance, as shown on subsection 3.3.

Post Processing Our final attempt to improve the results consisted in post processing the results. Since the model receives only a feature vector representing a candidate paragraph, it has no built-in schema for handling how the cases are structured and how many responses for each case would be reasonable. So it could classify 30 paragraphs as positive entailment instances for one case and identify none for another case. Analyzing the priors in the provided dataset (see 6), we can see that, on average, there are ~ 2 entailment paragraphs for each case. Thus we tried to post process the results and keep at most 5 entailment paragraphs for cases which had many candidates identified as true samples, and at least 1 paragraph for cases which had no candidates identified as true samples. To select which paragraphs to keep, we used the classifier confidence score. That provided a relevant increase in the model performance, especially for the precision score, as it is shown on table 7 (subsection 3.3).

3.3 Results

We detail below the results achieved by all variations of the method on the training dataset, and the three variations submitted on the COLIEE test dataset.

On the Training Dataset To evaluate the results on the training dataset, we set apart 25 cases from the original 181 cases (see Table 6). Those cases where not used on the development of our method (training classifiers, manual inspection, etc). Table 7 shows the results on that validation dataset.

Table 7. Results for the Case Law Entailment Task (validation dataset).

Algorithm	Precision	Recall	F-Measure
Word cloud	0.080	0.133	0.100
Similarity based	0.108	0.333	0.163
Similarity based + Word cloud	0.074	0.467	0.128
Similarity based + LIWC	0.238	0.167	0.196
Similarity based + Post processing	0.2424	0.2857	0.2622
Sim. based with Grad. Boosting + Data augmentation	0.7	0.25	0.3684

As a comparison baseline, we might consider a naive classifier which outputs all paragraphs as entailing the corresponding decisions, which would have the following scores in our validation dataset: 0.0291 (precision), 1.0 (recall) and 0.0428 (f-measure). The other naive classifier option (which would output false for all candidate paragraphs) would have a recall score of 0.0 and, as a consequence, an f-measure of 0.0, and thus is not a viable benchmark.

From Table 7, the best results our model attained were achieved using the similarity approach with a Gradient Boosting classifier and data augmentation (see subsection 3.2). However, we finished implementing it after the COLIEE dry run deadline, so we did not submit a formal result based on that approach. Not considering it, the best results were attained by combining the similarity based approach (with a Random Forest classifier) and post processing. That improved precision considerably and, despite a drop in recall, the F-measure had a steady improvement of 60.85% (from 0.163 to 0.2622). The word cloud approach showed poor results when used by itself, and even impaired the results of the similarity based approach when combined with it. Augmenting the similarity based approach with LIWC features provided a relevant increase of 20.24% on the F-measure. We did not have time to test post processing on that approach.

On the COLIEE Test Dataset Since we achieved the best results with the similarity based + post processing approach (see Table 7), we submitted three results based on it (acronym SB+PP), varying solely the number of estimators for the Random Forest. The results on the test dataset are shown on Table 8.

The best F-measure achieved was 0.258, which is still quite low and even somewhat disappointing. Nonetheless, that was the best performance among

Table 8. Results for the Case Law Entailment Task (real COLIEE test dataset).

Algorithm	Number of estimators	Precision	Recall	F-Measure
SB+PP	250 (default)	0.238095	0.283019	0.258621
SB+PP	100	0.190476	0.226415	0.206897
SB+PP	500	0.238095	0.283019	0.258621

all teams which submitted results for Task 2 in COLIEE 2018. The F-measure for the similarity based approach with a Gradient Boosting classifier and data augmentation was 0.12698, much lower than what we had on the validation dataset. That probably was due to a model over-fitting.

3.4 Future Work

The results presented above indicate this task has a high degree of difficulty, and shows the methods we used to tackle the problem must be refined. In fact, it may be more promising to devise completely new approaches. Applying word embeddings is still worthy for this task, but we need to improve the vector representations. We plan to use here the options cited on section 2.4: train a word embedding model with our own data, retrain the used model, test skip-gram and experimenting with other frameworks. It is worth trying to refine the use of the Gradient Boosting classifier, which achieved the best results on our validation dataset. With more time, we can investigate and configure the model to better generalize. However, we think the real gain in performance for Task 2 will come from a better understanding of the problem. We need to study what exactly makes an entailment relationship hold between case laws and then devise more effective strategies to model that relationship.

4 Statute Law Entailment

4.1 Method

For the statute law entailment, we used the characteristics that a legal statute consists of "general condition," "conclusion," "exceptional case," and "conclusion for the exceptional case." The query can belong to either "general condition" or "exceptional case." We filter out the cases in the statute that are not related to the input query, by figuring out where the input query belongs. We use condition/conclusion/exception detection rules defined in Kim et al. [22].

We determine if a query belongs to the general case or exceptional case by computing the proportion of shared words between query and condition, and query and exception_condition. We collect words from the detected condition (exception_condition) and conclusion (exception_conclusion), and also detect negation value for each condition (exception_condition) and conclusion (exception_conclusion): The negation level (neg_level(segment)) is computed as following: if [negation + antonym] occurs an odd number of times in the segment, its negation level is 1. If [negation + antonym] occurs an even (or zero)

number of times, its negation level is 0. We consider the entailment holds if neg_level(condition of the relevant article) + neg_level(conclusion of the article) is the same to neg_level(condition of the query) + neg_level(conclusion of the query). Otherwise, we consider the entailment does not hold. To detect negation and antonym, we manually constructed negation/antonym dictionary from the training data.

4.2 Experimental Results

The dry run data size of COLIEE 2018 is 644 queries for Task 4 from the Japanese bar exam from 2006 to 2016, and the formal run data size is 69 queries from the bar exam of 2017. Our performance of textual entailment is 63.77%, which was ranked No.1 in the COLIEE 2018 competition as shown in Table 9.

Table 9. Experimental results on formal run data of COLIEE 2018

Participating Team	Accuracy(%)
UA(our team)	0.6377
KIS_Frame	0.5652
KIS_mo3	0.5507
KIS_dict	0.5362
KIS_SVM	0.5217
KIS_Frame2	0.5072
UE	0.4783

5 Conclusion

The similarity based approach used for tasks 1 and 2 provided reasonable results on the COLIEE testing datasets. In Task 1, we ranked 4th place, whereas in Task 2 we ranked 1st place among all teams in COLIEE 2018. However, the absolute scores achieved show our method still needs to be improved before it could be used in real world applications. For the statute law competition, we proposed a method to answer yes/no questions from legal bar exams related to civil law statutes. We performed legal segmentation of the statute law, and used negation/antonym information for textual entailment. The performance of our system was ranked 1st place in Task 4 of the COLIEE 2018 competition.

Acknowledgments

This research was supported by Alberta Machine Intelligence Institute (AMII), and would not be possible without the further support of Colin Lachance from vLex and Compass, and the teams from Ross Intelligence and Intellicon.

References

1. Lee, M. D., Pincombe, B., Welsh, M.: An Empirical Evaluation of Models of Text Document Similarity. In Proceedings of the Annual Meeting of the Cognitive Science Society, vol. 27, pp 1254–1259 (2005).
2. Baeza-Yates, R. A., Ribeiro-Neto, B.: Modern Information Retrieval. Addison-Wesley Longman Publishing Co., Boston, MA, USA (1999).
3. Kumar, S.: Similarity Analysis of Legal Judgments and applying Paragraph-link to Find Similar Legal Judgments. Master thesis. Hyderabad, India (2014).
4. Chawla, N., Bowyer, K., Hall, L., Kegelmeyer, W.: SMOTE: Synthetic Minority Over-sampling Technique. Journal of AI Research, v. 16, 321–357 (2002).
5. Sun, Y., Kamel, M. S., Wong, A., Wang, Y.: Cost-sensitive boosting for classification of imbalanced data. Pattern Recognition, vol. 40 (12), 3358–3378 (2007).
6. Breiman, L.: Random Forests, Machine Learning, 45(1), 5–32 (2001).
7. Friedman, J. H.: Greedy Function Approximation: A Gradient Boosting Machine. Annals of Statistics, vol. 29, pp. 1189–1232 (2000).
8. Zhou, Z., Feng, J.: Deep Forest: Towards An Alternative to Deep Neural Networks. In Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, pp. 3553–3559 (2017).
9. Mikolov, T., Sutskever, I., Chen, K., Corrado, G., Dean, J.: Distributed Representations of Words and Phrases and Their Compositionality. In Proceedings of the 26th International Conference on Neural Information Processing Systems, vol. 2, pp 3111–3119, Lake Tahoe, NV, USA (2013).
10. Kusner, M., Sun, Y., Kolkin, N.I., Weinberger, K.: From word embeddings to document distances. In Proceedings of the 32nd International Conference on Machine Learning (ICML), pp 957–966 (2015).
11. Supreme Court of Canada, scc-csc.lexum.com. Accessed on Sep 4th, 2018.
12. Google Pre-trained Word Embeddings Model, https://code.google.com/archive/p/word2vec/. Last accessed on Sep 4th, 2018.
13. Pennebaker, J. W., Booth, R. J., Francis, M. E.: Linguistic Inquiry and Word Count: LIWC [Computer software]. Austin, TX: LIWC.net. (2007)
14. Tausczik, Y. R., Pennebaker, J. W.: The Psychological Meaning of Words: LIWC and Computerized Text Analysis Methods. Journal of Language and Social Psychology 29(1), pp. 24–54 (2010).
15. Online-Wörterbuch Englisch-Deutsch, www.dict.cc. Accessed on Sep 4th, 2018.
16. Pennington, J., Socher, R., Manning, C.: GloVe: Global Vectors for Word Representation. Empirical Methods in Natural Lang. Processing. pp. 1532–1543 (2014).
17. Bojanowski, P., Grave, E., Joulin, A., Mikolov, T.: Enriching word vectors with subword information. Transactions of the Association for Computational Linguistics, vol. 5, pp. 135146 (2017).
18. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. In Proc. Int. Conf. on Learning Representations (2013).
19. Manevitz, L. M., Yousef, M.: One-class Svms for Document Classification. The Journal of Machine Learning Research, vol. 2, pp. 139–154 (2002).
20. Le, Q., Mikolov, T.: Distributed Representations of Sentences and Documents. In Proc. of the 31st International Conf. on Machine Learning, Beijing, China (2014).
21. Valvoda, J., Ray, O.: From Case Law to Ratio Decidendi. New Frontiers in Artificial Intelligence. pp. 20–34. Springer International Publishing (2017).
22. Kim, M.-Y., Xu, Y., Lu, Y., and Goebel, R.: Question Answering of Bar Exams by Paraphrasing and Legal Text Analysis. In Lecture Notes in Artificial Intelligence (JSAI International Symp. on Artificial Intelligence), Vol.10247:299-313, (2017).

Case law retrieval with doc2vec and Elasticsearch*

Wilco Draijer and Suzan Verberne

Leiden Institute of Advanced Computer Science,
University of Leiden, Niels Bohrweg 1, Leiden, The Netherlands
w.j.w.draijer@umail.leidenuniv.nl
s.verberne@liacs.leidenuniv.nl

Abstract. In this paper a Random Forest Classifier with eight different features is implemented and evaluated as legal case retrieval system for COLIEE 2018. Based on a summary of and facts about a new case the system makes a prediction using TF–IDF, doc2vec and the More Like This score from Elasticsearch about the relevance of candidates. The system scored third out of twelve submissions in the competition. The precision is competitive, but the recall is fairly low. We found that TF–IDF and the More Like This score are the most important predictors of relevancy. Lowering the minimum document frequency of TF–IDF does not improve the results. As future work we propose noun phrase chunking as an extension of this system to retrieve the full name of multiple acts (e.g. 'the immigration and refugee protection act').

Keywords: Legal Information Retrieval · doc2vec · Elasticsearch

1 Introduction

The Competition on Legal Information Extraction/Entailment (COLIEE) [1] is an annual international competition on legal information retrieval. COLIEE 2018 exists of four separate tasks.

The first task involves case law retrieval. A number of supporting cases (on average ten) from 200 candidates are to be extracted for each of 285 new cases. The new cases have a short summary and corresponding facts. The candidates are not the same for every new case.

The second task is about entailment. A decision in a new case is based on a relevant old case. The aim of this task is to find the paragraph of the old case on which the decision is based.

The third task is finding relevant Japanese Civil Code Articles for a legal bar exam question. The answer is a number of articles for every question.

The fourth and final task is to identify relevant articles for questions.

This paper describes our participation in only the first task of the competition. The built system is a random forest classifier based on eight

* Supported by ORTEC – Optimize Your World

different features: four features based on TF–IDF, two features based on doc2vec, and two features based on a more-like-this query in Elasticsearch. We obtained satisfying results; our system ranked third out of twelve submissions from six different teams with an F1-score of 0.393.

The remainder of this paper is organized as follows: Section 2 gives background on the data the system needs to handle and provides a problem description. Section 3 describes the methods used to build the system. Section 4 describes the experiments we have done. Section 5 provides the results of the system, which are then discussed in section 6. Section 7 concludes the paper and makes suggestions for future work.

2 Data and Problem Description

For the case law retrieval task 285 new cases were given as training set. Each case has a textual summary describing the case, and a list of facts about the case. An example of a summary and facts of a new case is given in appendix A and B. The test set existed out of 59 new cases, also with 200 candidates.

Each of the new cases had 200 prior cases as candidates. The task at hand was to select for each new case which of these 200 candidates are relevant for the new case. There were between 3 and 16 relevant candidates for each case. The distribution of word counts of the candidate cases, the facts and the summary are somewhat skewed to the right side. This is shown in table 1, where the average, median and maximum word count, and standard deviation are given.

Table 1. Documents word count

Type	Median	Average	Std. Deviation	Maximum word count
Candidates	24,654	35,533	37,390.98	516,321
Facts	5,675	6,772	4,549.85	33,113
Summary	426	538	439.12	5,247

This shows that the candidates are significantly longer and presumably hold more information than the provided facts and summaries for the new cases.

3 Used Methods

Since classifiers are not able to handle text data directly, we have to engineer numeric features based on the text of the cases. If these numeric features are a good representation of the content of the cases, decision trees will be able to segregate the cases based on these numbers grouping similar cases together. A random forest classifier trains a number of decision trees, which will all vote on the classification of a case. The numeric features we want to extract from the candidate cases are based on three methods: TF–IDF, doc2vec and More Like This. In this section we will introduce all three methods.

3.1 TF–IDF

To find distinct characteristics of a document just counting words in a single document will not necessarily give a fair insight. Even after removing stop–words specific words appear very often in a document. If such words appear often in all documents it will not be a distinctive term. For example, in this task the word 'case' was frequently found in all candidates and summaries. This does not distinct any particular candidate.

To account for this we used Term Frequency – Inverse Document Frequency (TF–IDF) [2]. This metric indicates how often a word occurs in a document, weighted by the number of documents the word occurs in.

We use the implementation of the TfidfVectorizer in the sklearn package in Python. We create a TF–IDF vector for all documents to find similarity between documents. If the euclidean and/or cosine distance between vectors is small it shows that the documents have similar specific word counts which indicates that they have similar contents.

3.2 Doc2vec

Le and Mikolov [3] proposed an unsupervised framework inferring continuous distributed paragraph vector representations for documents, called doc2vec, which is an extension of the word2vec model [4]. Word2vec learns linguistic context of words by training neural networks with a large corpus of text. Doc2vec extends this by adding a paragraph ID to the vectors. Where word2vec learns a numeric representation of words, doc2vec learns a numeric representation of documents just by adding this paragraph ID. Doc2vec gained popularity over the last few years as it is useful for embedding larger blocks of text.

Similar documents will have similar doc2vec vector representations. Therefore we added a doc2vec vector to all documents and calculated the distance between the vectors to identify similarity.

3.3 More Like This

Elasticsearch [5] is an efficient search and analytics system. One of the functionalities in their search is the More Like This-query (MLT). It takes a text field, searches for similar texts and gives a score to all results. It uses the Lucene Scoring Formula [6], which uses TF–IDF. The Lucene Scoring formula is calculated as follows:

$$\begin{aligned} score(q,d) = {} & coord\text{-}factor(q,d) \cdot query\text{-}boost(q) \\ & \cdot \frac{V(q) \cdot V(d)}{|V(q)|} \cdot doc\text{-}len\text{-}norm(d) \cdot doc\text{-}boost(d) \end{aligned} \tag{1}$$

With:

– q as the query

- d as a document
- *coord-factor(q,d)*: When multiple items are queried, this factor may reward documents extra for every additional term matched
- *query-boost(q)*: the user can specify boosts to each query, sub-query, and each query term which will receive extra weight
- $V(q) \cdot V(d)$: the dot product of the weighted vectors (TF–IDF)
- $|V(q)|$: Euclidean norm
- *doc-len-norm(d)*: document length normalization factor
- *doc-boost(d)*: the user can specify important documents which will receive extra weight

The biggest difference between More Like This and the regular TF–IDF is that More Like This always normalizes on document length. Though relevance classification tends to favor longer documents, since they contain more words and therefore could be unfairly advantaged), documents holding absolute more relevant information could be more interesting than those which hold relatively more relevant information.

4 Experiments

The system that is built uses Elasticsearch as a database. Every case has the following different fields: original text, preprocessed text, TF–IDF distance score, doc2vec vector representation.

4.1 Preprocessing

To make the original text more usable multiple preprocessing steps are implemented.

1. Lowercased entire text
2. Removed additional info like the numbered tags for the paragraphs
3. Removed all signs and punctuation marks
4. Replaced 's. 4', 'sec. 4' or 'section 4' with 'section_4' (for all numbers and with or without dot after s or sec)
5. Step 4 but with 'art.' or 'article' and 'topic'
6. Replaced 'subject act' with 'subject_act'
7. Removed all stop words (using the nltk stop words package)
8. Removed all words with two or less characters
9. Lemmatized all words (using the nltk WordNetLemmatizer)

This preprocessing made the texts less complex, without removing essential words or phrases.

Table 2. Features

Features
More Like This Score on Facts
More Like This Score on Summary
Doc2vec Cosine Similarity distance to Facts
Doc2vec Cosine Similarity distance to Summary
TF–IDF Euclidean distance to Facts
TF–IDF Euclidean distance to Summary
TF–IDF Cosine similarity distance to Facts
TF–IDF Cosine similarity distance to Summary

4.2 Feature Extraction

We extracted eight features from all candidate cases. These features are shown in table 2.

TF–IDF is implemented for unigrams and a minimum document frequency of 400. We decided to create a new TF–IDF corpus for the test set and not use the one from the training set. The dimensionality of the TF–IDF vectorizer is 9,147 for the training set and 3,444 for the test set. Doc2vec is implemented with a vector size of 100. These features can be used as an input for a classifier.

4.3 Distance Metrics

TF–IDF and doc2vec are both vectors. The distance between two vectors can be measured in multiple ways. The most straightforward way is euclidean. This is a straight line distance measurement. Cosine similarity takes the angle between two vectors. These distances are shown in figure 1, where d is the euclidean distance and θ the cosine similarity.

When two documents have similar directions (topics), but not similar lengths (occurrences of certain words), the euclidean distance measure will be large while the cosine similarity not. If direction is a better indicator than length cosine similarity should be used. In this system the distance for doc2vec is measured with cosine similarity, TF–IDF is measured with cosine similarity and euclidean distance (as two different features).

4.4 Random Forest Classifier

The system uses a Random Forest Classifier, which uses the features described in table 2. It is trained and tuned with five fold cross validation to find the best parameters for the system. The chosen parameters based on this training are shown in table 3.

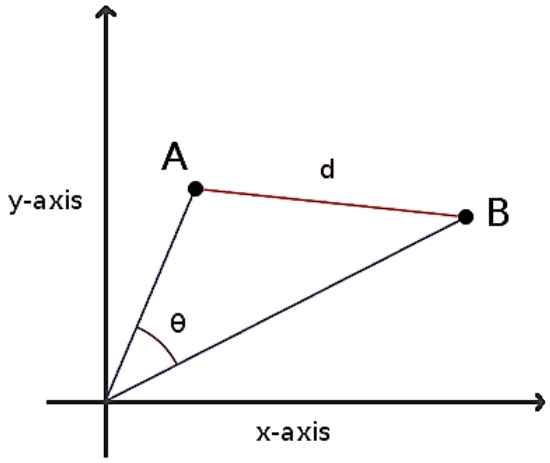


Fig. 1. Euclidean Distance (d) and Cosine Similarity (θ)

Table 3. Random Forest Classifier Parameters.

Parameter	*Value*	*Researched Range*
Number of trees	45	15–85 in steps of 5
Criterion	Gini (impurity)	Gini and Entropy
Minimum samples to split a node	2	2–12 in steps of 1
Maximum depth	35 nodes	10–80 in steps of 5
Minimum samples in a leaf	2	2–10 in steps of 1
Maximum features in a tree	Square root of number of features	1–8 in steps of 1 and square root
Cross-validation folds	5	3 and 5
Scoring methods	Precision, recall, f-measure	
Refit scoring methods	F-measure	

5 Results

The test data is classified by the system and COLIEE has sent the results of the classification back; these are shown in table 4, sorted by F-measure. Our team uses the id 'UL'.

Table 4. Results

id	*Precision*	*Recall*	*F-measure*
JNLP-k=10	0.676271	0.634340	0.654635
JNLP-r=2.5	0.546419	0.655008	0.595806
UL	**0.563798**	**0.302067**	**0.393375**
HUKB1	0.497436	0.308426	0.380765
UA-postproc	0.348422	0.403816	0.374080
UA-smote	0.353868	0.392687	0.372268
HUKB2	0.404661	0.303657	0.346957
UA	0.372477	0.322734	0.345826
Smartlaw	0.287076	0.430843	0.344565
UBIRLED-2	0.195511	0.720191	0.307536
UBIRLED-1	0.132881	0.623211	0.219056
UBIRLED-3	0.561404	0.101749	0.172275

5.1 Feature importance

The Random Forest Classifier attributed a feature importance to all features. These are shown in table 5.

Table 5. Feature Importance

Feature	Importance
More Like This Score on Facts	0.1868
TF–IDF Euclidean distance to Facts	0.1459
TF–IDF Cosine similarity distance to Summary	0.1452
TF–IDF Cosine similarity distance to Facts	0.1335
TF–IDF Euclidean distance to Summary	0.1116
More Like This Score on Summary	0.1103
Doc2vec Cosine similarity distance to Summary	0.0838
Doc2vec Cosine similarity distance to Facts	0.0829

This shows that a doc2vec vector representation of a document is less distinguishing than the other features.

5.2 Improvements

Based on these results we tried to improve the system by choosing a better minimum document frequency for TF–IDF, by grouping the train and test set in the corpus for TF–IDF, by shortening the length of the doc2vec vectors and by removing less important features. The tuning of the hyperparameters of the system using new or other features were researched the same way as shown in table 3 The results of the new improvements are shown in table 6.

We tried to improve the TF–IDF by decreasing the Minimum Document Frequency to 250, 40 and 4 and by grouping the corpora from the train and test set; improve doc2vec by decreasing the length to 50 from 100; and improve the system overall by removing doc2vec, since it is less distinguishing.

Table 6. Potential Improvements

Improvement	*Change/Value*	*Precision*	*Recall*	*F-measure*
Base line	-	0.5638	0.3021	0.3934
Minimum Doc. Freq.	4	0.5736	0.2353	0.3337
	40	0.5620	0.2448	0.3411
	250	0.5415	0.2798	0.3690
TF–IDF	Grouped corpora	**0.5970**	0.2496	0.3520
Doc2vec length	50	0.5253	0.2973	0.3797
Removing less important features	No Doc2vec	0.5537	**0.3196**	**0.4052**
	No eucl. and cosine to summary	0.4554	0.2925	0.3562

This shows that the F-measure does not improve with a reduced minimum document frequency; especially the recall reduces. Grouping the train and test set in the same TF–IDF corpus increases the dimensionality of TF–IDF to 10,365 and improved the precision a bit, but reduced the recall. Altering the doc2vec length to 50 has a slight decrease in F-measure.

When the doc2vec features are removed, the F-measure increases a bit. The new feature importance changes are shown in table 7. Since the euclidean and cosine distance to the summary are the new least important features, these were to be removed next. This results in a substantial decrease in F-measure.

5.3 Distance Importance

When the doc2vec features are removed, the system is in the end only based on TF–IDF with different distance measures. Since this seems to be redundant we research the impact of only using a subset of these features. Table 8 shows these results. This shows that though More Like This has a higher feature importance (especially the MLT score on Facts), it does not perform well when used without the other distance measures.

Table 7. New Feature Importance

Feature	Importance
More Like This Score on Facts	0.2364
TF–IDF Cosine similarity distance to Facts	0.1798
TF–IDF Euclidean distance to Facts	0.1692
More Like This Score on Summary	0.1433
TF–IDF Cosine similarity distance to Summary	0.1372
TF–IDF Euclidean distance to Summary	0.1342

Table 8. Distance Comparison

Distances used	*Precision*	*Recall*	*F-measure*
Cosine, Euclidean & Normalized Euc (MLT)	0.5537	0.3196	0.4052
Cosine & Euclidean	0.3741	0.3259	0.3483
Euclidean & Normalized Euc (MLT)	0.4886	0.3068	0.3770
Cosine & Normalized Euc (MLT)	0.5105	0.3100	0.3858
Only Cosine	0.4361	0.3307	0.3761
Only Euclidean	0.3839	0.3180	0.3478
Only Normalized Euc (MLT)	0.4000	0.2003	0.2669

6 Discussion

Compared to the other teams the results from table 4 are reasonably good. The precision is good, but the recall is relatively low. In the legal field recall is especially important, since retrieving relevant cases is more important than correctly classifying as irrelevant cases. To formulate it differently: a system that classifies all cases as irrelevant has a very good precision, but is completely useless.

The difference in feature importance is quite interesting. The two doc2vec features are less important, but the other features are not clearly separable by characteristic: cosine similarity is not more important than euclidean, or More Like This than TF–IDF. The Summary features do not all perform better or worse than the Fact features. There is also no distinction in using Euclidean or Cosine similarity, which results in no clear preference for using one over the other. Using a subset of distance metrics showed it was best to use all. The fact that the feature importances all are fairly equal implies that all TF–IDF and More Like This features are used as predictive characteristics.

6.1 Potential Improvements

The attempt to refine the F-measure by changing multiple settings resulted in one actual improvement. Removing the doc2vec features increased the F-measure a bit. Removing even more relatively unimportant features does not result in more improvement. The right length for a doc2vec vector differs per problem.

The results show that a length of 100 is closer to the right length than 50. Grouping the train and test set had a small negative impact on the result. This means that in this case training the Random Forest Classifier with one set and using is with another does not negatively impacts the system. This is presumably because only the distance between vectors matter, not about the actual vectors.

Using a subset of distance measures gives unexpected results. When using one metric More Like This scores the worst out of the three, but when using two the one without More Like This scores worse than the other two. In text mining and information retrieval the cosine distance is most popular. When using only one metric it is the best choice, as shown in table 8, but using all three gives a better result.

7 Conclusion

We proposed the system described in this paper for COLIEE 2018.

Overall the Random Forest Classifier works fine, although it leaves room for improvement. The precision of the system is good, but the recall is fairly low. The feature importance, table 5, shows that the More Like This score and the two distance metrics for TF–IDF are all similarly important for predicting relevancy of candidates. The doc2vec features were not a good indicator, the system performed better when they were removed.

The reason for the reasonable F-measure our system reached can be found in two distinctive things: the distance metrics and preprocessing. All our features are in the end based on TF–IDF (when removing doc2vec), but used different distance metrics (Euclidean, Cosine and More Like This used normalized Euclidean). We showed the system performs best when using all three, but when using one cosine is best to be used (as is the baseline in the field of text mining).

There are multiple options for improvement of the system. First, more features could be added. A topic model like LDA could be added, following the reasoning that similar documents will have similar topics. Second, multiple cases contain an act existing out of multiple words, e.g. 'the immigration and refugee protection act'. The current preprocessing changes this to 'immigration refugee protection_act'. To keep the act as a whole as one term noun phrase chunking should be applied. The downside of just applying this is losing some information, because if 'the immigration and refugee protection act' is changed to 'the_immigration_and_refugee_protection_act' it loses the individual words. This could be prevented by adding both the individual words as the whole noun phrase, which will increase the dimensionality of the vectors. More research is needed to investigate the value of these solutions.

References

1. COLIEE, https://sites.ualberta.ca/~miyoung2/COLIEE2018/. Last accessed 3 Sept 2018
2. Salton, G., Buckley, C.: Term-weighting approaches in automatic text retrieval. Information processing and management, **24**(5), 513–523 (1988)
3. Le, Q., Mikolov, T.: Distributed representations of sentences and documents. In International Conference on Machine Learning, 1188–1196 (2014, January)
4. Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., Dean, J.: Distributed representations of words and phrases and their compositionality. In Advances in neural information processing systems 3111–3119 (2013)
5. Elasticsearch, https://www.elastic.co/products/elasticsearch. Last accessed 3 Sept 2018
6. Apache Lucene Scoring Algorithm, http://lucene.apache.org/core/index.html, Last accessed 3 Sept 2018

A Summary Example

Bradley was a member of the armed forces who fell while showering on board his ship.
He applied under s. 21(2) of the Pension Act for benefits relating to upper back and neck injuries caused by the accident.
The Veterans Review and Appeal Board dismissed the claim because the accident was not sufficiently connected with military service as required by s. 21(2).
Bradley made a new claim related to low back pain caused by the same accident.
The Minister of Veterans Affairs indicated that, because Bradley's condition resulted from an accident that had previously been determined as not directly connected to service, she had no jurisdiction under s. 85(1) of the Pension Act to proceed with the application.
Bradley sought judicial review, requesting an order of mandamus to compel the Minister to determine the application.

B Facts Example

[1] Phelan, J.: This Applicant, a self-represented former member of the Armed Forces, has been locked in litigation over his disability claim for a slip in the shower of HMCS Qu'Appelle for several years and in this Court since 1999 (see Bradley v. Canada (Attorney General), [1999] F.C.J. No. 144). His first application for a pension based on upper back and neck injuries was finally settled in 2004.
[2] After the 2004 Federal Court decision, the Applicant started a second application for mechanical lower back pain based on the same incident. After further proceedings it was determined that the Department of Veterans Affairs was required to determine this new claim. It is the new claim which is the subject matter of this judicial review.
[3] The Applicant was undergoing officer training as an Acting Sub-Lieutenant aboard HMCS Qu'Appelle, a destroyer escort. The ship was alongside in Vancouver near the end of a training cruise in the Pacific. The ship was ultimately destined to its home port of CFB Esquimalt.
[4] The Applicant, having completed training for the day, went to the mess (presumably the Wardroom) where he had a few beers. The exact quantity was not established but the drinking took place on board a warship in a tightly regulated environment.
[5] Mr. Bradley contends that he slipped while in the shower cleaning up from his daily duties and prior to going ashore. He was found the next day in his bunk in severe pain. There was a notation in the medical file that he had significant alcohol in his system. There was some suggestion drawn that there was a connection between the shower incident and the consumption of alcohol on board; however, how that could have happened in the circumstances of an officer in training, in a wardroom, was not established.
[6] When the Department finally considered this new claim, it reached the conclusion that the fall in the shower did not arise out of, and was not directly connected with, the Applicant's military service. The Department made no further determination as to whether the fall caused the mechanical low back pain.
[7] A Review Panel affirmed the Department's decision to deny the Applicant pension entitlement.
[8] The Applicant appealed to the Appeal Board; however, on August 5, 2008, the Appeal Board affirmed the negative decision. It is this decision which is the subject of this judicial review.

Legal Information Retrieval by Association Rules

Ying Chen[1], Yilu Zhou[2], Zhen Lu[2], Hao Sun[2], Wenjun Yang[3]

[1] PayPal Inc.
[2] Gabelli School of Business, Fordham University
[3] LawCubes Technology Co. Ltd
yzhou62@fordham.edu

Abstract. With the explosive growth of legal documents, automatic Legal Information Retrieval(LIR) is critical to help law professionals to conduct quality services. In this paper, we propose using association rules to retrieval relevant legal documents. Our method is tested on three COLIEE 2018 tasks, and achieve acceptable and stable performance on all the tasks.

Keywords: Legal Information Retrieval (LIR), Association Rule, n gram, Machine Learning, Random Forest, COLIEE Competition.

1 Introduction

With the explosive growth of legal documents, automatic Legal Information Retrieval(LIR) is critical to help law professionals to conduct quality services. Due to long and complicate syntax structure of legal expressions, high abstract jargons, and limited datasets, deep learning based NLP approaches which are popular in general domains [15,16,17], perform poorly in legal domain. In this work, we propose an association-rule based framework for legal information retrieval. The contributions of our models:

- No supervised training is needed. Therefore, we can score and rank large amount of legal documents without manual labeling.
- The computational cost is low. The complexity to calculate association score is only O(n), make it practical to be used on online legal document search engines.
- With tailed optimization, the LIR performance of our method is good, which we will demonstrate using experiments on Competition on Legal Information Extraction/Entailment (COLIEE 2018) datasets.

This paper is structured as follows: In Section 2 we review the relevant work and propose our research questions in section 3. We propose our methodology in Section 4. In Section 5, we describe our experimental design, results, and discussions. We conclude the paper by summarizing our findings and contributions in Section 6.

2 Related work

Machine learning based NLP methods are still very popular in LIR researches. [2,3,5, 6, 7,8] use the Support Vector Machine (SVM) and Random Forests (RF) as classifiers, adopt semantic and syntax features, such as cosine similarity, n-gram intersection, various distance features, dependency bigram, negation similarity, paragraph links, paragraph link types, and etc. [6,9,10, 13] add lexical features such as TF-IDF score into machine learning classifiers to improve LIR performance. In addition, there are varies of works experiment on non-machine-learning methods in LIR domain. To automatically retrieval relevant articles for a given legal query, [7] uses K-means clustering, then calculates cosine similarity between queries and articles only in the same cluster. [1] calculates document similarity by BM25[14] with keyword expansion.

Researchers also ensemble different NLP methods to further improve retrieval efficiency. [4,9,12] ensemble similarity using least square method (LSM ensemble) and linear discriminant analysis (LDA ensemble). They also add Word2Vec and Latent semantic analysis (LSA) to further boost the overall accuracy. [11] proposes using a 2-stage method for LIR: 1) relevance analysis and 2) relevance disambiguation. In the relevance analysis stage, they calculate n-gram association index for each article. For the ones which have close or low relevance scores, the researchers further leverage term order probabilities for disambiguation.

We experiment both machine learning based method and association based method on LIR problems. The best performance is generated by association rule based method similar to [11], and we further enhance the accuracy. We believe our findings uncover valuable insights for future researches.

3 Legal Information Retrieval tasks—COLIEE

We research LIR methods within the context of the 5th Competition on Legal Information Extraction and Entailment (COLIEE-2018), which include tasks on both statute law and case law. We participant in three tasks in COLIEE 2018 competition: tasks 1, task 2, and task 3.

- Task 1 is a legal case retrieval task, given a new case d_q, require to retrieve relevant cases $\{d_{r1}, d_{r2}, ..., d_{rn}\}$ from a provided case law corpus.
- Task 2 is the legal case entailment task, given a case d_q and a relevant case d_r, require identification of paragraphs in d_q that entails the decision of case d_q.
- Task 3 is a statute law retrieval task, given a legal bar exam question d_q, require to retrieve relevant statutes $\{d_{r1}, d_{r2}, ..., d_{rn}\}$ from a provided database of civil code statutes.

Three tasks have similar structures, from which we derives the research question for this paper: given a text d_q, and a training dataset of (d_q, d_r) pairs, searching for relevant text $\{d_{r1}, d_{r2}, ..., d_{rn}\}$.

4 Proposed Approach

Since recent studies on LIR mostly leverage machine learning and linguistic similarity, we propose two sets of modeling frameworks to tackle our research question.

4.1 Machine Learning Based Model adopting Word2Vec/Doc2Vec as features (later refer as RF+Word2Vec model)

Machine learning based models are widely experimented in LIR domain. The most commonly used classifiers are Support Vector Machine (SVM) and Random Forest(RF). There are two challenges when using machine learning based methods on LIR problems: very limited training samples, and long/complex sentences (i.e., large amount of features). These two challenges make tree-based models be favored over other classifiers such as SVM and neural network(NN), because tree-based models only choose useful features while SVM and NN have to generate weights for all the features, the latter are much easier to cause overfitting. Therefore, in this paper, we choose to use RF as our machine learning classifier.

RF Tag Generation

Since LIR is not a natural regression/classification problem, we have to reconstruct the dataset before feeding it to the classifier. Given a document set $\{d_1, d_2, \dots, d_n\}$ and a relevancy pair set $P = \{(d_q, d_r)\}$. Denote all the source document $d_q \in Q$, $Q = \{d_{q_1}, d_{q_2}, \dots\}$. Denote all the target document $d_r \in R$, $R = \{d_{r_1}, d_{r_2}, \dots\}$. We construct the training dataset $\tilde{T}$ in 2-steps. We first cross-join all the elements in Q with all the elements in R, and remove all the records with $d_i = d_j$ $(d_i \in Q, d_j \in R)$. We assign tag 1 to those (d_i, d_j) pairs which can be found in the given relevancy pairs P, and assign tag 0 for all other records, then dataset T is generated:

$$T = \{T_k = (d_i, d_j, l_k) \quad | \, d_i \in Q, d_j \in R, d_i \neq d_j, l_k = \begin{cases} 1 & (d_i, d_j) \in P \\ 0 & otherwise \end{cases}\} \quad \text{--(1)}$$

Then we balance the dataset by keeping all the tag 1s and randomly down-sample all the tag 0s for each unique input d_q. The training dataset becomes $\tilde{T} \in T$ such that for $\forall d_q \in Q$, the number of set $\{(d_q, d_m, 1\} \in \tilde{T}, d_m \in R\}$ equals to the number of set $\{(d_q, d_n, 0\} \in \tilde{T}, d_n \in R\}$.

Below is an example demonstrating the dataset reconstruction process. Given a relevancy pair set $P = \{(d_q, d_r)\} = \{(\text{apple x}, \text{pear}), (\text{cat}, \text{dog})\} \mid d_q \in Q, d_r \in R$. We first cross-join all the elements in Q with all the elements in R, and remove all the records with $d_q = d_r$, then we have a dataset as Table 1. We assign tag 1 to those (d_q, d_r) pairs which can be found in the given relevancy pair set P, and assign tag 0 for all other records. Then we balance the dataset by keeping all the tag 1s and randomly down-sample all the tag 0 for each unique input d_q. The balanced dataset is as Table 2. We then generate Word2Vec/Doc2Vec features for all the records in the balanced dataset $\tilde{T}$.

Table 1. Reconstruct dataset T.

Q	R	Tag
apple x	pear	1
apple x	dog	0
apple x	cat	0
pear	apple x	1
pear	dog	0
pear	cat	0
dog	apple x	0
dog	pear	0
dog	cat	1
cat	apple x	0
cat	pear	0
cat	dog	1

→

Table 2. Balance training dataset $\tilde{T}$.

Q	R	tag
apple x	pear	1
apple x	dog	0
pear	apple x	1
pear	dog	0
dog	apple x	0
dog	cat	1
cat	pear	0
cat	dog	1

Word2Vec Features

Here is an example demonstrating how Word2Vec features are generated. Assuming there are two sentences d_{q1} and d_{q2}, unique numbers are used to represent each word.

d_{q1}: "How is it going?" → Vector $v_{d_{q1}}$: <1,2,3,4,5>.

d_{r1}: "I do not know how" → Vector $v_{d_{r1}}$: <6,7,8,9,1>.

The two vectors share a same element 1 because the original sentences share a same word "how". A Word2Vec dictionary can be constructed as follows:

WORD	How	is	it	going	?	I	do	not	know
VECTOR	1	2	3	4	5	6	7	8	9

In addition, we lemmatize the corpus in the preprocessing stage. For example, "am" "is" "are" are transferred to the word "be", and represented by a same vector. Python package Gensim is used to generate Word2Vec features. At the end of the process, each document is converted into an equal length vector ($V_{length} = N$). When record length shorter than N, the vector will contain ($N - V_{length}$) 0s at the end. When record length longer than N, the words after position N will be omitted. The length we pick is 500, which is longer than 98.4% of the records. The generated vector is our feature set.

Doc2Vec Features

Here is an example demonstrating how Doc2Vec features are generated. Assuming there are two documents d_{q1} and d_{q2}, close numbers are used to represent similar words.

d_{q1}: "He likes dog" → Vector $v_{d_{q1}}$: <-0.99,-0.51,0.71>.

d_{r1}: "She loves cat" → Vector $v_{d_{r1}}$: <-0.98,-0.53,0.72>.

Python package Gensim is used to generate Doc2Vec features. Users can either train their own models to define similar words, or rely on system-default movie model to define similar words. After words being converted to numbers, Singular Value Decomposition (SVD) is used to compress vector dimension to requested length. In the end,

the vector length does not represent word numbers, but represent the number of topics within each document.

With tags and features, the dataset $\tilde{T}$ can be directly used on RF classifiers. We use RF classifiers in TensorFlow to train. When scoring, we generate full cross-join dataset T on the test data without re-balancing. For each $d_q \in Q$, the model prediction scores of (d_q, d_r) are sorted in descending order, only top K d_r are output as relevant texts.

4.2 Association Rule Model

Machine learning methods have several disadvantages on LIR problems. Firstly, LIR problem usually have very limited training samples, which make machine learning models hard to converge. Secondly, when the sizes of Q and R expand, the space consumption of datasets and the training computational cost exponentially increase. To enhance the scalability of LIR solutions, we propose following association rule models.

Basic Association Rule Model (later refer as Basic-AR model)

BLEU (bilingual evaluation understudy)[18] algorithm evaluates the quality of machine-translation which transforms one natural language to another. We borrow the concept of BLEU score and propose association rule model to search for relevant texts.

Given a document set $\{d_1, d_2, \dots, d_n\}$, we transform d_q into n_gram set n_{d_q}, transform d_r into n_gram set n_{d_r}. Then the association score of (d_q, d_r) can be calculated using the following equation:

$$s_{d_{qi}d_{rj}} = \frac{\left|n_{d_{qi}} \cap n_{d_{rj}}\right|}{\left|n_{d_{qi}} \cup n_{d_{rj}}\right|} = \frac{\left|n_{d_{qi}} \cap n_{d_{rj}}\right|}{\left|n_{d_{qi}}\right| + \left|n_{d_{rj}}\right| - \left|n_{d_{qi}} \cap n_{d_{rj}}\right|} \quad \text{--(2)}$$

where $\left|n_{d_{qi}} \cap n_{d_{rj}}\right|$ is the number of n_grams within the insection of $n_{d_{qi}}$ and $n_{d_{rj}}$, and $\left|n_{d_{qi}} \cup n_{d_{rj}}\right|$ is the number of n_grams within the union of $n_{d_{qi}}$ and $n_{d_{rj}}$, $\left|n_{d_{qi}} \cup n_{d_{rj}}\right| = \left|n_{d_{qi}}\right| + \left|n_{d_{rj}}\right| - \left|n_{d_{qi}} \cap n_{d_{rj}}\right|$. For each $d_q \in Q$, we calculate its association score against all the $d_r \in R$, then sort the association scores of (d_q, d_r) in descending order, only top K d_r are output as relevant texts.

Below is an example demonstrating how to calculate associate score of (d_q, d_r).

d_{q1}: "a cat is on the mat." → bigram set $n_{d_{q1}}$: {a cat, cat is, is on, on the, the mat}.

d_{q2}: "i have a cat in my home." → bigram set $n_{d_{q2}}$: {i have, have a, a cat, cat in, in my, my home}.

d_{r1}: "there is a cat on the mat." → bigram set $n_{d_{r1}}$: {there is, is a, a cat, cat on, on the, the mat}.

d_{r2}: "a cat is in my home." → bigram set $n_{d_{r2}}$: {a cat, cat is, is in, in my, my home}.

The associate score of each (d_q, d_r) should be as Table 3. According to Table 3, for document d_{q1}, document d_{r1} is more relevant than document d_{r2}, while for document d_{q2}, document d_{r2} is more relevant than document d_{r1}.

Table 3. Association score example.

s	d_{r1}	d_{r2}
d_{q1}	3/8	2/8
d_{q2}	1/11	3/8

Supervised: Co-occurrence Association Rule Model (later refer as CoOccur-AR model)

Basic association rule model only considers the similarity between the source document and the target document, it does not leverage human-labeled relevancy dictionaries. To find out whether adding knowledge-based information helps or not on LIR problems, we propose an alternative association rule model.

Given a document set $\{d_1, d_2, \ldots, d_n\}$ and a relevancy pair set $P = \{(d_q, d_r)\}$, we transform d_q into n_gram set n_{d_q}, and d_r into n_gram set n_{d_r}. Define an n_gram pair set between each n_gram w_k within set n_{d_q} and each n_gram w_l within set n_{d_r} to be: $NP_{qr} = \{(w_k, w_l) \mid w_k \in n_{d_q}, w_l \in n_{d_r}\}$. Denote NN to be the set which contains all the n_grams within n_{d_q} and n_{d_r}: $NN = \bigcup_{d_k \in Q \cup R} n_{d_k}$. For $\forall w_k \in NN, \forall w_l \in NN$, we construct a co-occurrence association matrix which count the frequency of (w_k, w_l) appear as a pair in the relevancy pair set P. We also record the term frequency as $\#d_k$ when $n_k = n_l$.

$$M(w_k, w_l) = \begin{cases} \#\{NP_{qr} \mid (w_k, w_l) \in NP_{qr} \mid (w_l, w_k) \in NP_{qr}\} & n_k \neq n_l \\ \#\{d_k \mid w_k \in d_k, (d_k \in Q \mid d_k \in R)\} & n_k = n_l \end{cases} \quad \text{--(3)}$$

Finally, we normalize the co-occurrence association matrix by dividing its term frequency.

$$\widetilde{M}(w_k, w_l) = \frac{M(w_k, w_l)}{M(w_k, w_k)} \quad \text{--(4)}$$

For each $d_q \in Q$, we calculate its association score $(\sum \widetilde{M}(w_k, w_l) \mid w_k \in n_{d_q}, w_l \in n_{d_r})$ against all the $d_r \in R$, then sort the association scores of d_r in descending order, only top K d_r are output as relevant texts.

Below is an example demonstrating how to construct co-occurrence associate matrix and calculate co-occurrence associate score for (d_q, d_r). Assume the relevancy pair set $P = \{(d_q, d_r)\} =$

$\{(\text{apple x}, \text{pear}), (\text{cat}, \text{dog}), (\text{apple y}, \text{dog}), (\text{apple x}, \text{pear})\} \mid d_q \in Q, d_r \in R$.

If we only consider unigram, we can count the number of times when each unigram w_k appears together with each unigram w_l. As to the diagonal of the matrix, we records the number of times each unigram appears in the relevancy pair set P. The co-occurrence association matrix M should be as Table 4.

Table 4. Co-occurrence association matrix M.

Co-occurrence	apple	pear	cat	dog	x	y
apple	3	2		1		
pear	2	2			2	
cat			1	1		
dog	1		1	2		1
x		2			2	
y				1		1

Then we normalize the co-occurrence association matrix, dividing each row by its diagonal value (i.e., number of times each unigram appears in P). The purpose for normalization is to remove the effect of unbalanced term frequency in the texts.

Table 5. Normalized co-occurrence association matrix $\widetilde{M}$.

Normalized Co-occurrence	apple	pear	cat	dog	x	y
apple	1	2/3		1/3		
pear	1	1			1	
cat			1	1		
dog	1/2		1/2	1		1/2
x		1			1	
y				1		1

Then for the document "Apple x" $\in Q$, we calculate its co-occurrence association score ($\sum \widetilde{M}(w_k, w_l) \mid w_k \in n_{d_q}, w_l \in n_{d_r}$) against all the $d_r \in R$:

(apple x, pear): 5/3

Co-occurrence	pear
apple	2/3
x	1

(apple x, dog): 1/3

Co-occurrence	dog
apple	1/3
x	

(apple x, cat): 0

Co-occurrence	cat
apple	
x	

(apple x, apple y): 1

Co-occurrence	apple	y
apple	1	
x		

According to the co-occurrence association scores, the relevancy of (apple x, pear)> (apple x, apple y)>(apple x, dog)>(apple x, cat).

5 Experiments and Results

We experiment our LIR models on three tasks within COLIEE 2018 competition: tasks 1, task 2, and task 3.

5.1 Task 1: The Legal Case Retrieval Task

Experimental Setup

Task 1 is a legal case retrieval task, given a new case d_q, require to retrieve relevant cases $\{d_{r1}, d_{r2}, ..., d_{rn}\}$ from a provided case law corpus.

In this task, the size of Q is 286, the size of R is 200. The relevancy pair P is also given, but for each document $d_q \in Q$, the number of its relevant cases specified in P varies from 1 to 53, as shown in Fig 1. We remove common English stop words and feed the documents into our LIR models for experiments.

Parameter Adjustment

- For the RF-Word2Vec model, we reconstruct the balanced training dataset, then use Gensim Doc2Vec [19] to generate 50 numbers long vectors from (Q,

R) for each record. 30% of the training dataset is reserved for validation. The balanced training dataset includes 5628 records, tested depths of random forest include [5, 20, 30, 50, 100], 50 performs best on the validation dataset, and tree_number=100.

- For the Basic-AR model, we tried {1~4}_gram, and find bigram performs best in this case. we further calculate document frequency (DF) for each term t as follows:

$$DF_t = \frac{1}{k}\sum_k \frac{number\ of\ times\ t\ appear\ in\ d_k}{total\ number\ of\ terms\ in\ d_k} \quad \text{--(5)}$$

d_k is the kth document. We remove the terms with document frequency larger than 0.02, for those frequently appear terms can be consider as stop words. However, we do not set the lower bound of document frequency, because legal terms are very domain-dependent, un-common terms can have great prediction power in finding relevant legal documents.

- In addition, we ensemble the RF-Word2Vec and the Basic-AR models, and test its prediction power. To be specific, we treat the AR_score as a feature, and feed it into the RF-Word2Vec model for classification. We later refer this experimental setting as RF-Word2Vec+AR.

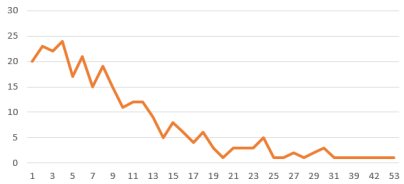


Fig 1. Number of relevant cases for each case in Task 1.

Table 7. Task 1: Basic-AR performance on test dataset

id	F-measure	Precision	Recall
HUKB1	0.38	0.50	0.31
HUKB2	0.35	0.40	0.30
JNLP-r=2.5	0.60	0.55	0.66
JNLP-k=10	0.65	0.68	0.63
Smartlaw	**0.34**	**0.29**	**0.43**
UA	0.35	0.37	0.32
UA-postproc	0.37	0.35	0.40
UA-smote	0.37	0.35	0.39
UBIRLED-1	0.22	0.13	0.62
UBIRLED-2	0.31	0.20	0.72
UBIRLED-3	0.17	0.56	0.10
UL	0.39	0.56	0.30

Evaluation Method

We adopt the evaluation method of COLIEE Task 1 to measure the performance of our LIR models.

$$Precision = \frac{\text{(the number of correctly retrieved cases(paragraphs) for all queries)}}{\text{(the number of retrieved cases(paragraphs) for all queries)}}$$

$$Recall = \frac{\text{(the number of correctly retrieved cases(paragraphs) for all queries)}}{\text{(the number of relevant cases(paragraphs) for all queries)}}$$

$$F - measure = \frac{\text{(2 x Precision x Recall)}}{\text{(Precision + Recall)}} \quad \text{--(6)}$$

Results

On the validation dataset, Basic-AR Model performs best among the three models. The best f-measure is achieved when output top 9 cases as relevant cases.

Table 6. Task 1: Basic-AR performance on validation dataset

Top n	AR			RF			RF+AR		
	F-measure	**precision**	**recall**	**F-measure**	**precision**	**recall**	**F-measure**	**precision**	**recall**
1	0.17	0.54	0.1	0	0.02	0	0	0.04	0
2	0.25	0.5	0.16	0.01	0.02	0.01	0.03	0.04	0.02
3	0.3	0.47	0.22	0.01	0.03	0.01	0.03	0.06	0.02
4	0.33	0.44	0.26	0.02	0.02	0.01	0.03	0.04	0.02
5	0.35	0.42	0.3	0.02	0.03	0.02	0.05	0.06	0.04
6	0.37	0.4	0.34	0.02	0.03	0.02	0.05	0.06	0.04
7	0.37	0.38	0.37	0.02	0.03	0.02	0.05	0.06	0.04
8	**0.38**	0.36	0.39	0.02	0.02	0.02	0.04	0.04	0.04
9	**0.38**	0.35	0.42	0.02	0.02	0.02	0.04	0.04	0.04
10	**0.38**	0.34	0.44	0.02	0.02	0.02	0.04	0.04	0.04
11	**0.38**	0.33	0.46	0.02	0.02	0.03	0.05	0.04	0.06
12	**0.38**	0.32	0.47	0.02	0.02	0.03	0.05	0.04	0.06
13	0.37	0.3	0.49	0.03	0.02	0.03	0.05	0.04	0.06
14	0.37	0.29	0.51	0.03	0.02	0.03	0.05	0.04	0.06
15	0.37	0.28	0.52	0.03	0.02	0.03	0.05	0.04	0.06
16	0.36	0.28	0.53	0.03	0.02	0.03	0.05	0.04	0.06
17	0.36	0.26	0.54	0.03	0.02	0.04	0.05	0.04	0.08
18	0.35	0.26	0.55	0.03	0.02	0.04	0.05	0.04	0.08

The Basic-AR model performance holds well on test dataset (Table 7). And our method is with low cost, scalable, and have stable performance over different datasets.

5.2 Task 2: The Legal Case Entailment task

Experimental Setup

Task 2 is the legal case entailment task, given a case d_q and a relevant case d_r, require identification of paragraphs in d_q that entails the decision of case d_q.

In this task, the size of Q is 181. The relevancy pair P is also given, but for each document $d_q \in Q$, the number of its relevant paragraphs specified in P varies from 1 to 4, as shown in Fig 4.

Parameter Adjustment

- For the RF-Word2Vec model, we reconstruct the balanced training dataset, then use Gensim Doc2Vec [19] to generate 50 numbers long vectors of (Q, R) for each record. 30% of the training dataset is reserved for validation. The balanced training dataset includes 478 records, the tested depths of random

forest include [5, 20, 30, 50, 100], 20 performs best on the validation dataset, and tree_number=100.

- For the Basic-AR model, we tried {1~4}_gram, and find bigram performs best in this case. In this task, We do not remove the terms with high document frequency, because after removing terms with high document frequency, prediction performance drops. Similar to task 1, we do not set the lower bound of document frequency.
- In addition, we ensemble the RF-Word2Vec and the Basic-AR model, and test its prediction power.

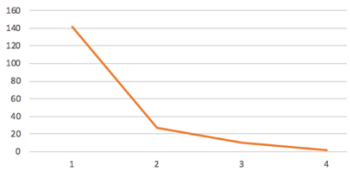


Fig 4. Number of relevant paragraphs for each case in Task 2.

Table 9. Task 2: Basic-AR performance on test dataset (ID: Smartlaw)

id	F-measure	Precision	Recall
Smartlaw	**0.07**	**0.05**	**0.15**
UA	0.26	0.24	0.28
UA-100estimators	0.21	0.19	0.23
UA-500estimators	0.26	0.24	0.28
UBIRLED-1	0.09	0.05	0.83
UBIRLED-2	0.09	0.05	0.92
UBIRLED-3	0.09	0.05	0.79
UNCC0	0.04	0.03	0.06

Evaluation Method

Model performance on task 2 is evaluated using the same F-measure as task 1, which listed in equation (6).

Table 8. Task 2: Basic-AR performance on validation dataset

Top n	AR			RF			AR+RF		
	F-measure	**Precision**	**Recall**	**F-measure**	**precision**	**recall**	**F-measure**	**precision**	**recall**
1	0.07	0.07	0.07	0	0.01	0	0.00	0.02	0.00
2	0.07	0.06	0.11	0.01	0.01	0.02	0.03	0.02	0.04
3	**0.09**	0.06	0.16	0.02	0.01	0.03	0.03	0.02	0.06
4	**0.09**	0.06	0.2	0.02	0.01	0.04	0.03	0.02	0.08

Results

On the validation dataset, Basic-AR Model performs best. The best f-measure keep increasing when K increases. Because in grant truth, all the cases have less than 4 relevant paragraphs, we select top K=4 as relevant paragraphs and do not go beyond 4. The Basic-AR model performance holds well on the test dataset (Table 8).

5.3 Task 3: The Statute Law Retrieval Task

Experimental Setup

Task 3 is a statute law retrieval task, given a legal bar exam question d_q, require to retrieve relevant statutes $\{d_{r1}, d_{r2}, ..., d_{rn}\}$ from a provided database of civil code statutes.

In this task, the size of Q is 657, and the size of R is 1056. The relevancy pair P is also given, but for each document $d_q \in Q$, the number of its relevant cases specified in P varies from 1 to 7, as shown in Fig 7. We remove common English stop words and feed the documents into our LIR models for experiments.

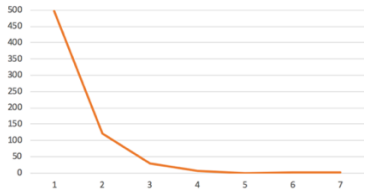


Fig 7. Number of relevant articles for each query in Task 3.

Parameter Adjustment

- For the Basic-AR model, we experiment on {1~4}_gram, and find bigram+trigram performs best in this case. We remove the terms with document frequency larger than 0.01, for those frequent appear terms can be consider as stop words. We do not set the lower bound of document frequency, for uncommon terms can have great prediction power in finding relevant articles.
- For the RF-Word2Vec model, we reconstruct the balanced training dataset, then use Gensim Word2Vec to generate 500 numbers long vectors of (Q, R) for each record. 30% of the training dataset is reserved for validation. The balanced training dataset includes 1380 records, the tested depths of random forest include [5, 20, 30, 50, 100], 30 performs best on the validation dataset, and tree_number=1000. Since RF+AR performs better than RF in both task 1 and task 2, we ensemble the RF-Word2Vec and the Basic-AR model in task 3, to test its prediction power.
- For the CoOccur-AR model, we experiment on {1~4}_gram, and find trigram performs best in this case.

Evaluation Method

Model performance on task 3 is evaluated using a different F2-measure comparing to task 1 and task 2. This evaluation matrix focus more on recall than precision.

$$Precision = \text{average of } \frac{(\text{the number of correctly retrieved articles for each query})}{(\text{the number of retrieved articles for each query})}$$

$$Recall = \text{average of } \frac{(\text{the number of correctly retrieved articles for each query})}{(\text{the number of relevant articles for each query})}$$

$$F2 - measure = \frac{(5 \text{ x Precision x Recall})}{(4 \text{ x Precision } + \text{ Recall})} \quad --(7)$$

Table 10. Task 3: 3 model performance on validation dataset

Top n	F2-measure			Recall			Precision		
	AR	**Co-AR**	**RF+AR**	**AR**	**Co-AR**	**RF+AR**	**AR**	**Co-AR**	**RF+AR**
1	0.48	0.23	**0.20**	0.47	0.23	0.19	0.53	0.26	0.23
2	**0.49**	0.25	**0.20**	0.51	0.29	0.20	0.43	0.17	0.18
3	**0.49**	**0.26**	0.19	0.54	0.34	0.21	0.36	0.13	0.15
4	0.48	0.24	0.19	0.56	0.36	0.22	0.31	0.11	0.13
5	0.47	0.24	0.19	0.57	0.39	0.23	0.28	0.09	0.12
6	0.46	0.23	**0.20**	0.58	0.41	0.24	0.25	0.08	0.11
7	0.45	0.21	**0.20**	0.59	0.42	0.26	0.23	0.07	0.10

Table 11. Task 3: Basic-AR performance on test dataset (ID:smartlaw)

lang	sid	F2 meas-ure	Preci-sion	Recall	MAP	R_5	R_10	R_30
E	UB3	0.70	0.78	0.69	0.80	0.80	0.85	0.96
J	HUKB2	0.69	0.77	0.68	0.78	0.79	0.84	0.93
J	HUKB1	0.68	0.75	0.68	0.78	0.79	0.84	0.93
J	ORGJ1	0.66	0.74	0.65	0.77	0.78	0.84	0.93
?	YA	0.66	0.72	0.65	0.75	0.73	0.75	0.85
E	ORGE1	0.64	0.71	0.63	0.74	0.75	0.81	0.90
E	UB2	0.62	0.68	0.62	0.75	0.80	0.87	0.96
E	JNLP1	0.61	0.41	0.71	0.74	0.76	0.82	0.92
E	**smartlaw**	**0.60**	**0.41**	**0.70**	**0.70**	**0.71**	**0.76**	**0.83**
E	JNLP2	0.60	0.41	0.70	0.73	0.75	0.81	0.91
E	SPABS_bm25	0.58	0.40	0.67	0.71	0.78	0.82	0.91
E	UE	0.45	0.49	0.45				
E	smartlaw_3gram	0.44	0.49	0.43	0.47	0.45	0.46	0.51
E	UB1	0.42	0.45	0.41	0.54	0.57	0.72	0.82
?	smartlaw_2gram	0.34	0.30	0.43	0.46	0.44	0.48	0.52
E	SPABS_rnnen	0.22	0.14	0.25	0.26	0.34	0.45	0.57
E	SPABS_rnnsq	0.20	0.12	0.23	0.27	0.35	0.45	0.61

Results

On the validation dataset, Basic-AR model performs best among the three models. Basic-AR and RF-Word2Vec models achieve the best f-measure when output top 2 articles as relevant articles, while CoOccur-AR model achieves the best f-measure when output top 3 articles as relevant articles. The Basic-AR model performance holds well on test dataset (Table 11). On the test dataset, Basic-AR model performance is even better.

5.4 Discussion

We experiment our three LIR models on task 1, task 2, and task 3 within COLIEE 2018 competition. The Basic-AR achieve the best performance over RF-Word2Vec and CoOccur-AR on all the three tasks, which suggests unsupervised and non-machine-learning based method have great potential on LIR problems. In addition, Basic-AR model is with low cost, scalable, and have stable performance over different datasets, it can save both human efforts and computational costs on legal information retrieval tasks.

We have tried several attempts to further tune the performance of Basic-AR. Firstly, instead of using (single-n) gram, we found (multi-n) gram can generate better performance. In task 3, it has been proved that bigram+trigram achieves the best prediction performance. Secondly, in this paper, we only try to output static top K d_r as output. For example, in task 3, we find the optimal K to be 2, then for all the queries d_q, we only consider top 2 article d_r as relevant articles. However, some queries may have only 1 relevant article while others may have 3 relevant articles. Static top K outputs may not satisfy such need. We experiment on using Basic-AR score as threshold for cut off, but the performance is not as good as static top K. Future researches is required to find a better relevancy cut off.

6 Conclusion

In this paper, we propose three models to tackle LIR problems, and we experiment on three COLIEE 2018 tasks. The association based model outperform the rest two models, and achieve stable and acceptable performance on all three COLIEE tasks. Therefore, we demonstrate non-machine learning method has great potential on LIR problems.

References

1. Daiki Onodera and Masaharu Yoshioka, “Civil Code Article Information Retrieval System based on Legal Terminology and Civil Code Article Structure”, Tenth International Workshop on Juris-informatics (JURISIN), 2016 (Submission ID: HUKB)
2. Truong-Son Nguyen, Viet-Anh Phan, Thanh-Huy Nguyen, Hai-Long Trieu, Ngoc-Phuong Chau, Trung-Tin Pham, and Le-Minh Nguyen, “Legal Information Extraction/Entailment Using SVM-Ranking and Tree-based Convolutional Neural Network”, Tenth International Workshop on Juris-informatics (JURISIN), 2016 (Submission ID: JNLN2)
3. Danilo S. Carvalho, Vu Duc Tran, Khanh Van Tran, Viet Dac Lai, and Minh-Le Nguyen, Lexical to Discourse-level Corpus Modeling for Legal Question Answering”, Tenth International Workshop on Juris-informatics (JURISIN), 2016 (Submission ID: JNLN1)
4. 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 (Submission ID: iLis7)
5. Phong-Khac Do, Huy-Tien Nguyen, Chien-Xuan Tran, Minh-Tien Nguyen and Nguyen Le Minh, “Legal Question Answering using Ranking SVM and Deep Convolutional Neural Network”, Tenth International Workshop on Juris-informatics (JURISIN), 2016 (Submission ID: JNLN3)

6. Mi-Young Kim, Ying Xu, Yao Lu and Randy Goebel, “Legal Question Answering Using Paraphrasing and Entailment Analysis”, Tenth International Workshop on Jurisinformatics (JURISIN), 2016 (Submission ID: UofA)
7. Adebayo Kolawole John , Luigi Di Caro, Guido Boella, and Cesare Bartolini, “TEAMNORMAS' PARTICIPATION AT THE COLIEE 2016 BAR LEGAL EXAM COMPETITION”, Tenth International Workshop on Juris-informatics (JURISIN), 2016 (Submission ID: N01)
8. Yoshioka, M., & Onodera, D. (2017). A Civil Code Article Information Retrieval System based on Phrase Alignment with Article Structure Analysis and Ensemble Approach. In 4th Competition on Legal Information Extraction and Entailment (COLIEE 2017), 16th International Conference on Artificial Intelligence and Law (ICAIL 2017) .
9. Heo, S., Hong, K., & Rhim, Y.-Y. (2017). Legal Content Fusion for Legal Information Retrieval. In the 16th International Conference on Artificial Intelligence and Law (ICAIL 2017).
10. Nguyen, T.-S., Phan, V.-A., & Nguyen, L.-M. (2017). Recognizing entailments in legal texts using sentence encoding-based and decomposable attention models. In 4th Competition on Legal Information Extraction and Entailment (COLIEE 2017), 16th International Conference on Artificial Intelligence and Law (ICAIL 2017) .
11. Carvalho, D. S., Tran, V., Tran, K. Van, & Nguyen, L. M. (2017). Improving Legal Information Retrieval by Distributional Composition with Term Order Probabilities. In 4th Competition on Legal Information Extraction and Entailment (COLIEE 2017), 16th International Conference on Artificial Intelligence and Law (ICAIL 2017).
12. Nanda, R., John, A. K., Caro, L. Di, Boella, G., & Robaldo, L. (2017). Legal Information Retrieval Using Topic Clustering and Neural Networks. In 4th Competition on Legal Information Extraction and Entailment (COLIEE 2017), 16th International Conference on Artificial Intelligence and Law (ICAIL 2017) .
13. Kim, M.-Y., & Goebel, R. (2017). Two-step Cascaded Textual Entailment for Legal Bar Exam Question Answering. In the 16th International Conference on Artificial Intelligence and Law (ICAIL 2017) .
14. S. E. Robertson and S. Walker. Okapi/Keenbow at TREC-8. In Proceedings of TREC-8, pages 151{162, 2000.
15. Hamid Palangi, Li Deng, Yelong Shen, Jianfeng Gao, Xiaodong He, Jianshu Chen, Xinying Song, and Rabab Ward. Deep sentence embedding using long short-term memory networks: Analysis and application to information retrieval. IEEE/ACM Trans. Audio, Speech and Lang. Proc.,24(4):694{707, April 2016.
16. Guido Zuccon, Bevan Koopman, Peter Bruza, and Leif Azzopardi. Integrating and evaluating neural word embeddings in information retrieval. In Proceedings of the 20th Australasian Document Computing Symposium, ADCS '15, pages 12:1{12:8, New York, NY, USA, 2015. ACM.
17. Debasis Ganguly, Dwaipayan Roy, Mandar Mitra, and Gareth J.F. Jones. Word embedding based generalized language model for information retrieval. In Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '15, pages 795-798, New York, NY, USA, 2015. ACM.
18. Papineni, K.; Roukos, S.; Ward, T.; Zhu, W. J. (2002). BLEU: a method for automatic evaluation of machine translation (PDF). ACL-2002: 40th Annual meeting of the Association for Computational Linguistics. pp. 311–318. CiteSeerX 10.1.1.19.9416 .
19. Rathore, M. (2018). gensim doc2vec & IMDB Sentiment Dataset. Retrieved from https://markroxor.github.io/gensim/static/notebooks/doc2vec-IMDB.html

HUKB at COLIEE2018 Information Retrieval Task

Masaharu Yoshioka[1,2] and Zihao Song[1]

[1] Graduate School of Information Science and Technology, Hokkaido University, N14 W9, Kita-ku, Sapporo-shi, Hokkaido, Japan
yoshioka@ist.hokudai.ac.jp
[2] Global Station of Big data and cybersecurity, Hokkaido University, N14 W9, Kita-ku, Sapporo-shi, Hokkaido, Japan

Abstract. In COLIEE-2018, there are four tasks as a combination of document domains (case law and statute law) and task types (information retrieval (IR) and entailment) and we (HUKB) participated in the IR tasks of two document domains (Tasks 1 and 3). In this paper, we introduce our system for the submission of final results and discuss the characteristics of the system.

Keywords: Legal Information Retrieval · Query Analysis · Query Construction.

1 Introduction

In Competition on Legal Information Extraction/Entailment (COLIEE)-2018, there are four tasks as a combination of document domains (case law and statute law) and task types (information retrieval and entailment). In this COLIEE-2018 [3], we (HUKB) participated in the information retrieval (IR) task of two document domains (Tasks 1 and 3).

In this paper, we introduce our system for the submission of final results. For Task 1, we mainly discuss the method of handling structural information of case law for query and database construction. For Task 3, we mainly discuss the method of identifying difficult questions to use different approaches for selecting relevant articles. We also discuss the characteristics of the system by using the evaluation results of the training data and test data.

2 Case Law Retrieval Task

2.1 System architecture

Because the case law retrieval task (Task 1) was newly introduced, there was no general approach of constructing an IR system for the task.

In this framework, we built an IR system for the task by using the following two steps to retrieve referred cases from the database.

[3] https://sites.ualberta.ca/ ~ miyoung2/COLIEE2018/

1. Ranked retrieval
   We use an IR system to generate ranked retrieval results.
2. Selection of referred cases
   By using the ranked retrieval results, we select the final answer from them.

For the ranked retrieval results, we conducted experiments to address the following criteria.

1. How to handle structural information of case law.
   Case law texts consist of multiple parts (summary, fact, citation, and others). We discuss which part of the text is useful to make the index and query.
2. Two database construction methods to use candidate documents for IR.
   In the training data, there were 200 similar candidate case documents for each query. However, to obtain better term distribution information among all case laws, it is preferable to have a larger text database. We constructed two databases; one was constructed of candidate documents only, and the other of all candidate documents. For the latter database, the final results were selected from candidate documents.

For the selection of referred cases, we propose the following two simple methods.

- Fixed number selection
  The system returns a fixed number of cases for all queries. The number of returned cases is decided based on the preliminary experiments using test data.
- Variable number selection
  The system selects related cases by checking the similarity with unrelated cases. In this experiment, we assumed that cases for all other queries were unrelated cases.

2.2 Case law text analysis

The text structure of case law is not well formalized, but most of the text has a similar style. We extracted structural information based on pattern matching of a section, such as lines in the text.

First, we classified the text into three parts, namely header, facts, and footer. The beginnings of the facts are identified by a fact representative line that starts with number 1 in brackets (i.e., `[1]`). The end of the facts part is identified by the lines `Editor:` or `[End of document]`.

From each part, we extracted the following information.

- Header parts
  Header parts contain a variety of information, such as reference information, name of the court, summary, counsel, solicitors of record, etc. We extracted the summary information by checking the line that starts with `Summary:` and extracted text until the line with other heading keywords

(e.g., `Administrative Law`, `Statues Noticed`, and `Counsel`). If the system fails to extract summary parts from the header, the system uses all header text of a case as a summary.
- Fact parts:
  Several documents had facts part in two languages (English and French). In most of the cases, the French parts were followed by English and the same numbers were used. So, we excluded text parts after the appearance of the second `[1]`.
- Footer parts:
  We did not use footer parts for the experiment.

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

To discuss which parts of the texts are useful to identify related cases, we used three different text parts to construct a document database, as follows.

- $Part_{sum}$: Summary only
- $Part_{fact}$: Fact only
- $Part_{all}$: All (summary and fact).

In addition, because query texts were given as `summary.txt` and `fact.txt`, we could construct three types of queries as follows.

- Q_{sum}: `summary.txt`
- Q_{fact}: `fact.txt`
- Q_{all}: `summary.txt` and `fact.txt`.

2.3 Database construction with two different document sets

The original IR task was designed to select referred case documents from 200 candidate documents. Although there is no information on how to select 200 such documents from all the case documents, the statistical information of the terms (e.g., document frequency, term collocation information) for these candidates might not be the same as that for all case laws. As a result, the statistical information extracted from candidate documents only may have problems (e.g., when candidate documents are selected from the existence of important keyword(s) in the query, the inverted document frequency (IDF), such as importance measures of those terms, are discounted compared with the case with all case documents).

Therefore, we constructed two databases for IR.

- DB_{each}: The database for the IR system was constructed by using candidate documents only (we constructed a database for each query).
- DB_{all}: The database for the IR system was constructed by using all documents. For training analysis, we used documents for training data only, but for the final submission, we used all documents (training and test) for database construction.

For the latter case, because IR results that use DB_{all} may return documents that are not included among the candidates, we applied postprocessing to select candidate cases for the query from the result.

2.4 Experimental analysis using training data

First, we discuss how to generate ranked retrieval results with two discussion criteria (usage of structural information and database construction with different document sets) by using a preliminary experiment with 18 different parameter settings (3 Parts × 3 Queries × 2 DBs).

We used Indri for the basic IR system with out-of-the-box settings. Indri is a state-of-the-art IR system based on language modeling [3] [4] and the retrieval performance of Indri is also good for COLIEE Statute Law retrieval with the same settings [4].

Table 1 shows the mean average precision (MAP) of the retrieved results with different parameter settings (3 Parts × 3 Queries × 2 DBs).

Table 1. MAP of different parameter settings

Text type and Query type	DB_{each}	DB_{all}
$Part_{all}$+Q_{all}	0.3616	0.3852
$Part_{all}$+Q_{sum}	0.4044	**0.4284**
$Part_{all}$+Q_{fact}	0.3498	0.3732
$Part_{sum}$+Q_{all}	0.3488	0.4020
$Part_{sum}$+Q_{sum}	0.3591	0.3981
$Part_{sum}$+Q_{fact}	0.3298	0.3827
$Part_{fact}$+Q_{all}	0.3567	0.3801
$Part_{fact}$+Q_{sum}	0.3972	0.4240
$Part_{fact}$+Q_{fact}	0.3437	0.3684

From these results, we confirmed that $Part_{all}$+Q_{sum}+DB_{all} were the best results from among these settings. In addition, $Part_{all}$+Q_{sum}+DB_{all} significantly outperformed other settings statistically (Wilcoxon signed rank test p = 0.01), except for $Part_{fact}$+Q_{sum}+DB_{all}. The following summarizes this experiment.

- It is better to use a larger document set (DB_{all}) to extract statistical information of the terms.
- It is better to use summary text (Q_{sum}) for query construction. Summary text may cover all important keywords and fact text may contain unnecessary keywords that are not appropriate for the query.
- It is better to use all text ($Part_{all}$) for text database. Using summary text only may miss keywords that exist in the query. Although the performance with using fact only ($Part_{fact}$) is close to that using all text ($Part_{all}$), it may be reasonable to use all text.

After this experiment, we conducted all experiments using the $Part_{all}$+Q_{sum}+DB_{all} setting.

For the selection of referred cases, we conducted the following experiment.

[4] https://www.lemurproject.org/indri/

– Fixed number selection
  Based on the preliminary experiment, we generated a list with a fixed number of returned cases. From comparison, we selected eight for fixed number selection.
– Variable number selection
  We assume that the related cases are highly ranked with other cases; therefore, when the number of the related cases is small, the number of cases that belong to other candidate cases increases. Therefore, the number of cases for the question that exists in the highly ranked retrieved results may decrease. Therefore, when we control the number of documents to check whether the document belongs to the candidates, the number of cases selected from the list is controlled. In this experiment, we introduced two parameters; one is the minimum number of cases $Cand_{min}$ that should be included for the retrieval results, and the other is the maximum number of cases $Check_{max}$ that are checked to verify whether documents belong to the candidates. When the checked retrieved results reach $Check_{max}$ and the number of cases to select is below $Cand_{min}$, the system checks the list to find $Cand_{min}$ cases using the ranked list. We conducted experiments with different parameter settings using the test data and decided that $Cand_{min} = 3$ and $Check_{max} = 40$ in this experiment.

2.5 Submitted results

We submitted two runs that use a variable number selection setting (HUKB1) and a fixed number selection setting (HUKB2).

Table 2. Evaluation results of submitted runs and best run (Task 1)

run	number of returned documents	precision	recall	F
HUKB1	390	0.4974	0.3084	0.3808
HUKB2	472	0.4047	0.3037	0.3469
JNLP-k=10		0.6763	0.6343	0.6546

From this result in Table 2, since HUKB1 has a higher evaluated value for both precision and recall, we confirm that it is better to have a mechanism that controls the number of returned cases for each query. However, the performance of the system is not as good as the best performance system (JNLP-k=10).

However, there is no explanation about how to construct a candidate case set. If it is difficult to construct such candidates automatically, the result of HUKB1 might not be reproducible. In such a case, it is necessary to consider other methods for controlling the number of returned cases.

3 Statute Law Retrieval Task

3.1 System architecture

We already participated in the statute law retrieval task of COLIEE [2, 5] and our system for the statute law IR task is an extension of the previous system.

The following is the basic system architecture of our previous system [5].

1. The system uses the output of IR results with different settings as a feature for final rank calculation using SVM rank [1].
2. Train the ranking model with different settings and merge the results based on their outputs.

Differences from the previous system are as follows.

1. Manual construction of database of conditions and decisions for civil law articles. We manually checked all civil law articles to extract the condition parts (for judicial decisions) of the articles and their judicial decisions to construct an article database of the condition and decision parts. The system extracts the condition and judicial decision parts from the question and uses them as queries to calculate similarity between questions and articles.
2. Using Indri [3] instead of BM25.
   As we confirmed, Indri achieved good performance even for out-of-the-box settings [4]; we use Indri for a basic IR engine to calculate the similarity score among queries and articles. To make an index of Indri, we used the MeCab morphological analyzer to split the sentence into a word sequence. We used the same procedures for splitting questions into query word sequences. Details of these procedures are explained in [4].
3. Introducing a new system to identify topic difficulty.
   Based on the analysis of the performance of all COLIEE participants [4], there are two types of queries: easy queries for which almost all participants can select at least one related candidate article for the query, and difficult queries for which almost all participants cannot retrieve related articles. We introduce a machine learning-based system to identify such topic difficulty.

Details of these differences are discussed in the following subsections.

3.2 Extraction of condition and decision parts

As the articles use multiple sentences and enumerations to describe the conditions of the judicial decisions, it is not very easy to classify this information using a simple syntactic analysis. Therefore, we manually checked all the articles of Japanese civil law to select the condition and judicial decision parts.

Compared with the articles, it is not very difficult to extract the condition and judicial decision parts from the questions using the dependency parser KNP [5] and clue keywords. 1 shows an output of KNP for the question in the training

[5] http://nlp.ist.i.kyoto-u.ac.jp/EN/index.php?KNP

data (H18-26-1). ”ＡＢが甲建物を持分各２分の１の割合で共有していた場合，Ａが死亡して相続人も特別縁故者もいないときは，甲建物の所有権はＢに帰属する。” (”In cases where person A and person B co-own building X at the ratio of 1:1, if person A dies and had no heirs or persons with special connection, the ownership of building X belongs to person B.”) that belongs to the former case. From this dependency tree, we select the condition part branches by using keywords such as ”場合 (in case)”, and ”とき (if)”. In this case, we select branches that end in ”場合 (in case)”, and ”とき (if)” (rounded rectangular with thick dotted line) and select ”ＡＢが甲建物を持分各２分の１の割合で共有していた場合，Aが死亡して相続人も特別縁故者もいないときは，(In cases where person A and person B co-own building X at a ratio of 1:1, if person A dies and had no heirs or persons with special connection)” (rectangular with thin dotted line) as a condition part and ”甲建物の所有権はＢに帰属する。(ownership of building X belongs to person B.)” as the decision part.

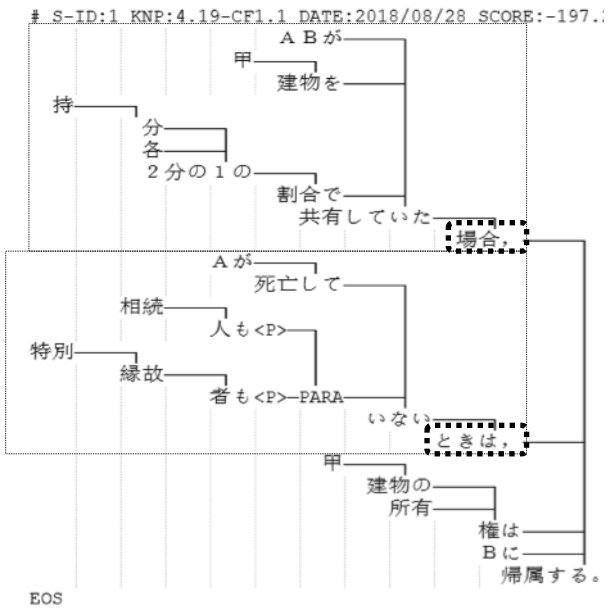


Fig. 1. Result of KNP parsing

3.3 Ranked retrieval system using SVM rank

This year’s system uses Indri for generating ranked retrieval results as features of SVM rank.

We used the following features for SVM rank.

1. Indri scores with various pairs of queries and text parts
   We constructed three types of queries (condition part, decision part, all text). We also constructed three document databases that contain a part of a document (condition, decision, title, and contents) and one document database with all text (title and contents). We calculated Indri-based similarity scores for combinations of all queries and all document databases except the condition query with the decision database and the decision query with the condition database) ($3 \times 5 - 2 = 13$ similarity scores are calculated).
2. The longest common subsequence with various pairs of queries and text parts
   For the same 13 variations, scores based on the longest common subsequence proposed in the previous system [5] were calculated.

As a result, we have 26 features for calculating similarities between questions and articles. All feature values were normalized to the 0–1 scale using maximum and minimum values using linear transformation (maximum values correspond to 1 and minimum values correspond to 0).

To generate different variations of models for checking the difficulty of the problem, we constructed 11 variations of models for SVM rank by excluding one year data (H19?H28, H18+H20-H28, ..., and H18-H27) for training data. In this year, we use the SVM rank software [6] based on scikit-learn. [7]

In addition, we also used each model to check the difficulty of the question. We conducted a retrieval for cross validation setting (e.g., evaluated H18 questions using a model that excludes H18 data for training) and checked the rank of all the relevant articles. When the ranks were larger than 50, we classified these results as outliers for training the model to retrieve easier questions. The following are difficult questions that were excluded as outliers H18-23-I, H18-23-U, H18-27-U, H18-29-4, H19-15-I, H21-24-E, H22-15-A, H22-27-2,H24-19-1, H24-22-3, H24-27-A, H24-30-3, H25-15-4, H25-21-O, H26-14-I, H26-16-E, H26-27-C, H26-27-I, H26-32-G, and H28-14-1. In most cases, these questions require knowledge about mapping between keywords. For example, in the case of H18-23-I ”Aが宝石をBに売り，その代金をBがCに支払うとの契約を締結し，Cが受益の意思表旨をした場合，Aが宝石をBに引き渡したが，Bが代金をCに支払わないときは，CはBに対して代金を自己に支払うよう請求することができるが，AもBに対して代金をCに支払うよう請求することができる。”(”In cases where person A sold a jewel to person B, person B entered into a contract where person B was assigned to pay the purchase money to person C, and person C expressed his/her intention to enjoy the benefit of the contract, if person A delivered the jewel to person B, and person B has not paid the purchase money to person C yet, while person C can demand that person B should pay the purchase money, person A can also demand that person B should pay the purchase money to person C.”), it is necessary to understand that ”支払う (pay)” and ”意思表旨 (express intention)” of the question corresponds to ”給付 (tender)”, and ”約する (promise) in the articles to retrieve article 537. We assume that these queries

[6] https://gist.github.com/agramfort/2071994
[7] http://scikit-learn.org/

are difficult questions discussed in [4] and decided to treat them as outliers for training the SVM rank model for retrieving articles for easy questions.

After identification of such outliers, we regenerated the SVM rank model by excluding such questions from the training data.

By using the output of 11 models, we obtained 11 different ranking results for each question. Then, the system calculated the average rank of each article and sorted the articles using this average rank in ascending order to generate final merged retrieval results.

3.4 Identification of difficult questions

The identification of difficult questions is a task to estimate the outliers discussed in the previous section without using answer information.

Since topics become difficult when there is term mismatch among questions and articles, we assume that the similarity measure for first-ranked documents of the retrieved results may be comparatively worse than for easy ones.

Therefore, we used the document similarity score information for the first-ranked document (i.e., Indri scores and longest common subsequence-based scores with various pairs of queries and text parts are used for the features to identify the topic difficulty and linear SVM of scikit-learn is used to classify the topic difficulty).

For the difficult questions, to include the relevant articles in the list, we decided to include 100 articles to keep a better recall.

Another perspective of difficulty can be checked by the consistency among the retrieval results of the 11 models. There were several cases in which the 11 models selected differed in their top-ranked articles. In such a case, it may be better to include all top-ranked articles to increase the recall of the system.

3.5 Submission results

We submitted two different versions for Task 3. One is a method that considers the difficulties of questions (HUKB1) and the other selects top-ranked documents only (HUKB2).

As the system cannot identify the difficult questions by using SVM, the results of HUKB1 and HUKB2 are almost the same except for the inclusion of extra articles because of the inconsistency among the 11 models (five questions have two articles for the retrieved results).

Table 3 shows the evaluation results of the submitted runs. As HUKB1 and HUKB2 use the same merged rank results, the MAPs of these two runs are the same. As the added secondary ranked document using inconsistency among the 11 models did not include any related articles, the performance of HUKB1 was worse than HUKB2. ORG1 used Indri, which used all the text of questions as query and all parts of articles for constructing the text database. The result of ORG1 was almost identical to HUKB2, except that HUKB2 found relevant articles for H29-3-O and H29-6-E. For these two cases, ORGJ1 found the relevant document as second-ranked, but HUKB2 found it as first-ranked.

Table 3. Evaluation results of submitted runs (Task 3) and corresponding organizers' run

run	return	retrieved	precision	recall	F2	MAP
HUKB1	74	53	0.7162	0.5955	0.6163	0.7479
HUKB2	69	53	0.7681	0.5955	0.6235	0.7479
ORGJ1	69	51	0.7391	0.5730	0.6000	0.7398

From these results, our system improved slightly compared with simple Indri, but the difference was rather marginal.

3.6 Discussion

Despite the effort to improve the system performance using several techniques, the final performance of the system is currently not very impressive.

There are several issues to be solved before discussing the appropriateness of this approach.

1. Revision of the system to extract conditions and decisions
   Based on the analysis of the final results of extracting conditions and decisions of the test data, there were several questions where the system failed to extract the appropriate parts. The quality of extraction needs to be improved and the results checked after the revision.
2. Analysis of difficult questions
   Evaluation of the appropriateness of the feature set used for difficult question identification is needed. Possible candidates for the additional features are ones calculated by the score difference between the first-ranked documents with other documents (e.g., second-ranked and tenth-ranked). If the first-ranked documents are closely similar to the question (e.g., longer longest common sequence), the scores of the first-ranked documents are comparatively higher than that of the second- or tenth-ranked documents. In contrast, if there is a smaller overlap between the first-ranked documents and the question, the difference becomes smaller than in the former case.

However, if there are several bugs in the system that generate the final submission results, the system needs to be revised to check the performance based on the new results.

4 Summary

In this paper, we introduced our system for participating in Tasks 1 and 3 of COLIEE-2018. For the Task 1 system, we found that it was better to use larger case data to estimate term statistical information and the summary part was more important than the fact part to construct the query. We also confirmed that the number of cases needs to be controlled. For Task 3, there was no significant difference compared with the basic IR system. The appropriateness of the system needs to be checked to decide whether the proposed approach is promising.

Acknowledgement

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

References

1. Joachims, T.: Training linear svms in linear time. In: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 217–226. KDD '06, ACM, New York, NY, USA (2006). https://doi.org/10.1145/1150402.1150429, http://doi.acm.org/10.1145/1150402.1150429
2. Onodera, D., Yoshioka, M.: Civil code article information retrieval system based on legal terminology and civil code article structure. In: The Proceedings of the 10th International Workshop on Juris-Informatics (JURISIN2016) (2016), paper 19
3. Strohman, T., Metzler, D., Turtle, H., Croft, W.B.: Indri: A language model-based search engine for complex queries. In: Proceedings of the International Conference on Intelligent Analysis. pp. 2–6 (2005)
4. Yoshioka, M.: Analysis of coliee information retrieval task data. In: Arai, S., Kojima, K., Mineshima, K., Bekki, D., Satoh, K., Ohta, Y. (eds.) New Frontiers in Artificial Intelligence. pp. 5–19. Springer International Publishing, Cham (2018)
5. Yoshioka, M., Onodera, D.: A civil code article information retrieval system based on phrase alignment with article structure analysis and ensemble approach. In: Satoh, K., Kim, M.Y., Kano, Y., Goebel, R., Oliveira, T. (eds.) COLIEE 2017. 4th Competition on Legal Information Extraction and Entailment. EPiC Series in Computing, vol. 47, pp. 9–22. EasyChair (2017). https://doi.org/10.29007/5zzj, https://easychair.org/publications/paper/PRH

Legal Question Answering System using FrameNet

Ryosuke Taniguchi[1], Reina Hoshino[1] and Yoshinobu Kano[1]

[1] Informatics Course, Graduate School of Integrated Science and Technology,
Shizuoka University, 3-5-1 Johoku, Naka-ku, Hamamatsu City, Japan
rtaniguchi@kanolab.net rhoshino@kanolab.net
kano@inf.shizuoka.ac.jp

Abstract. A central issue of yes/no question answering is the usage of knowledge source given a question. While yes/no question answering has been studied for a long time, legal yes/no question answering largely differs from other domains. The most distinguishing characteristic is that legal issues require precise analysis of predicate argument structures and semantical abstraction in these sentences. We have developed a yes/no question answering system for answering questions for a statute legal domain. Our system uses a semantic database based on FrameNet, which works with a predicate argument structure analyzer, in order to recognize semantic correspondences rather than surface strings between given problem sentences and knowledge source sentences. We applied our system to the COLIEE (Competition on Legal Information Extraction/Entailment) 2018 task. Our frame based system achieved better scores on average than our previous system in COLIEE 2017, and was the second best score among participants of Task 4. We confirmed effectiveness of the frame information with the COLIEE training dataset. Our result shows the importance of the points described above, revealing opportunities to continue further work on improving our system's accuracy.

Keywords: COLIEE, Question Answering, Legal Bar Exam, Legal Information Extraction, FrameNet.

1 Introduction

Automatic question answering is attracting more interests recently. Due to the increasing expectation to the Artificial Intelligence (AI) technologies, people tend to regard question answering systems as a brand new technology emerged today. However, most successful systems employ rather traditional techniques of question answering which have decades of history [1] [2] [3] [4] [5] [6] [7], including series of shared tasks such as TREC [8], NTCIR [9] and CLEF [10]. This paper describes our challenge to the COLIEE 2018 legal bar exam, which asks participants to answer true or not based on the civil law Articles, given text drawn from the Japanese legal bar exam.

A variety of algorithms and systems has been proposed for question answering. Typically, these question answering systems used *big data* for answering questions [11] [12] [13] [14]. For example, Dumais et al. [15] focused on the redundancy available in

large corpora as an important resource. They used this redundancy to simplify their algorithm and to support answer mining from returned snippets. Their system performed quite well given the simplicity of the techniques being utilized.

The now widely known IBM Watson system [16] would be considered as a typical example of such a question answering system of the big data approach. The IBM Watson system won in the Jeoperdy! Quiz TV program competing with human quiz winners. The core Watson system employed a couple of open source libraries, including the traditionally well-designed DeepQA system [17] as its skeleton of question answering processing. Because their target domain, the Jeoperdy! Quiz, could ask broad range of questions, they collected a huge amount of knowledge sources from the Internet, etc., extracting relevant knowledge by combining a couple of different natural language processing (NLP) techniques.

Answering university examinations is another example. The Todai Robot project [18] is a challenge to solve Japanese university examinations, focusing towards attaining a high score in the National Center Test for University Admissions by 2016, and passing the entrance exam of the University of Tokyo (Todai) in 2021 [19]. Although the Todai Robot project tries to achieve higher scores, their aim is rather to reveal the current performance and limitation of the existing AI technologies, using the examinations as its benchmark, similar to the COLIEE's legal bar exam task. In contrast to the COLIEE task, the challenge of Todai Robot project includes variety of subjects including Mathematics, English, Japanese, Physics, History, etc. all written in Japanese language. While solving any problem of these subjects could be considered as question answering, some problems require special technologies. For example, Mathematics and Physics require to process formula; Japanese requires to infer emotions of story characters. Solving the History subjects might be considered as rather an extension of the existing question answering issues. The Todai Robot project achieved better scores than the average of the real human applicants in their Mock Exam challenges.

Recognition of textual entailments (RTE or RITE) is another related issue. RTE has been intensively studied for recent days, including shared tasks such as RTE tasks of PASCAL [20][21], SemEval-2012 Cross-lingual Textual Entailment (CLTE) [22], NTCIR RITE tasks [23][24][25], etc. In the third PASCAL RTE-3 task, contradiction relations are included in addition to entailment relations [21]. In the RTE-6 task, given a corpus and a set of candidate sentences retrieved by a search engine from that corpus, systems are required to identify all the sentences from among the candidate sentences that entail a given hypothesis. NTCIR-9 RITE, NTCIR-10 RITE2, and NTCIR-11 RITEVal Exam Search tasks [25] required participants to find an evidence in source documents and to answer a given proposition by yes or no. Research of RTE normally tries to employ logical processing.

As described above, question answering techniques could include logic, reasoning, syntactic and semantic analysis. Many previous related works tried to employ such deeper analyses. However, required techniques more or less differ depending on a target domain. Another issue is whether the knowledge source needs to be "big data" or not. Regarding the COLIEE's legal problems, required knowledge source can be limited.

In this paper, we suggest using a semantical corpus based on a Rule-based predicate argument structure analyzer in a precise way, rather than to use any machine learning

methods. Due to this small data issue, supervised machine learning methods would suffer from insufficient training data. In addition, there are no "similar" problems for most of the legal bar exam problems. Therefore, a solver needs to "comprehend" the contents of the knowledge sources. Moreover, it is difficult to analyze why machine learning answers so, due to their black box architecture. Rule-based methods would make analyses less difficult and are especially effective in a limited domain like legal documents.

Based on these thoughts, we built our yes/no question answering system. Our system does not employ any machine learning. The main method of our system is a predicate argument structure analyzer using FrameNet. We integrated them and applied to COLIEE 2018 Task 4. Our frame based system achieved the second best score among participants. We compared our frame based system with our previous system as baselines, confirming effectiveness of the frame information. There are still many difficult issues remained to be solved though.

We explain about previous works and FrameNet in Section 2. Section 3 describes our design of the yes/no question answering system especially using FrameNet. Section 4 shows our experimental results for this COLIEE task and the comparison with previous system. We discuss our achievements and limitations comparing with previous system in Section 5, mentioning possible future works in Section 6. We conclude our paper in Section 7.

2 Background

2.1 COLIEE

The COLIEE shared task series is held in association with the JURISIN (Juris-informatics) workshop. The first one was the COLIEE 2014 shared task [26]. Following this, COLIEE 2015 shared task [27], COLIEE2016 shared task [28], and COLIEE 2017 [29] shared task (this time in conjunction with ICAIL) were held. This paper mainly describes our participation to the COLIEE 2018 shared task. We call COLIEE 2018 simply as COLIEE in this paper.

The COLIEE shared task consists of four tasks. Task 1 is the legal case retrieval task which involves reading a new case and extracting related cases. Task 2 is the legal case entailment task which compares the new case with related cases given by Task 1.

Task 3 of this legal question answering task involves reading a Japanese legal bar exam question and extracting a subset of Japanese Civil Code Articles. Task 4 is a legal question answering task which requires both of the legal information retrieval system and textual entailment system. Given a set of legal yes/no questions, a participant's system will retrieve relevant civil law articles. Then, answer yes/no entailment relationship between input yes/no question and the retrieved articles.

2.2 Previous Work

In COLIEE 2016 [30] , our yes/no question answering system was based on case-role analyses using JUMAN [31] and KNP [32]. JUMAN is a Japanese morphological analyzer where we added a custom dictionary for legal technical terms based on a Japanese legal term dictionary (“有斐閣法律用語辞典第4版”). KNP is a Japanese dependency case structure analyzer, works on top of JUMAN. Using results of these tools, we obtained a subject and an end-of-sentence expression for each sentence. A subjective case is normally specified by particles “が” or “は”, which are subjective case markers in Japanese. We regarded these cases as subjective cases. When we analyzed the civil law articles, we removed each header part “X条 (Article X)”, which includes an article name and numbers. We compared the pairs of the subject and the end-of-expression between the civil law articles and the legal bar exams.

Our COLIEE 2017 [33] system was based on our COLIEE 2016 system above. We defined our own *clause* unit (“節”) in order to recognize condition clauses and proposition clauses precisely, which are included in a single sentence. After recognizing condition clauses and proposition clauses in a sentence, we compared corresponding clauses between a given question and civil law articles. A clause should include a predicate as a core element of that clause. We applied a dependency parser that makes chunks (“文節”) of a couple of morphemes. Starting form a chunk that includes a predicate, we aggregated neighboring chunks when a neighboring chunk does not include any predicate.

As comparing clauses, we used three modules, a precise match, a loose match and a rough match. The precise match performed exact matches for its predicate, its subject and its object. When we could not find any subject nor any object, we skip that sentence. We outputted yes if everything matched, else outputted no. When there was any negation either in problem clause or in article clause, we reversed the yes/no output. The loose match was looser version of the precise match. When comparing proposition clauses and condition clauses, we outputted yes if either subjects or objects were match in addition to matching predicates. The rough match was the loosest match. We only compared predicates of proposition clauses.

2.3 FrameNet

FrameNet [34] [35] is an English semantical lexical database based on a theory of meaning called frame semantics [36]. Basic idea of frame semantics is that people understand the meaning of words largely by frames which they evoke. Frames have some semantic roles called Frame Elements (FEs). Frame evoking words are called Lexical Units (LUs). For example, a typical situation of shopping involves *buyers*, *sellers*, *goods*, *money*, *means*, *rate*, and *unit*. FrameNet has ten kinds of relations (*Inheritance*, *Using*, *Perspective_on*, *Subframe*, *Precedes*, *Inchoactive_of*, *Causative_of*, *Metaphor*, *See_also*, and *ReFraming_mapping*) within frames (called Frame Relations). Fig. 1 shows an example of the “Commerce-transaction” frame evoked by the shopping concept in FrameNet. There is a Japanese version of FrameNet [37]. We use LUs of Japanese FrameNet, in addition to the English version of FrameNet.

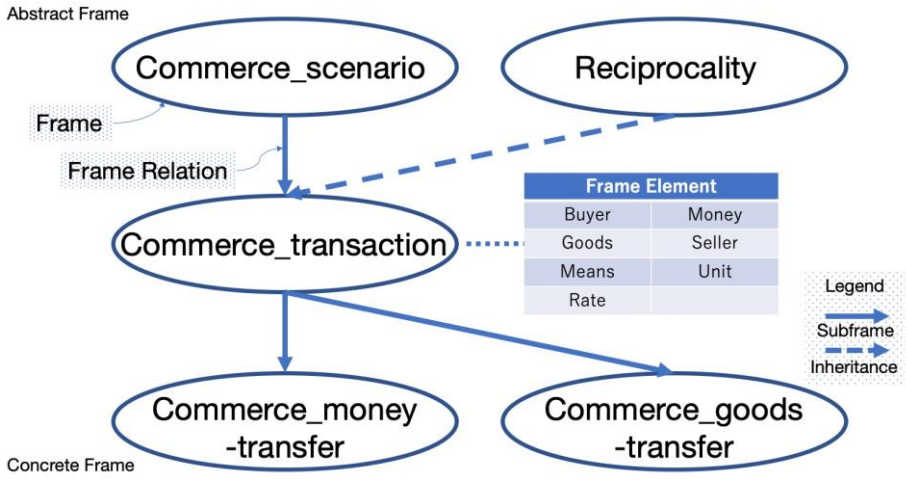


Fig. 1. An example shows a "Commerce-transaction" frame. Each node shows a frame, and an arrow between two frames shows a frame relation.

We use Japanese WordNet [38], in addition to FrameNet. Japanese WordNet is a lexical database for Japanese, where synonyms, hypernyms, hyponyms and English translation words are defined. We use WordNet to expand lexical units of FrameNet.

3 Proposed Method

We use FrameNet in addition to the previous rule-based system. A reason is that structures of civil law articles are clean. The civil law articles use only one place (snippet) for one topic. While our previous system performed textual entailments in a superficial layer, our proposed method using FrameNet could perform in a deeper level. Another reason is that we need precise analyses to solve legal issues, rather than statistically calculate rough estimate values in a superficial way. We took an unsupervised approach for the same reasons.

3.1 Previous Rule-based System

We define two types of clauses in our previous system: proposition clauses and condition clauses. Before using FrameNet, we obtain these clauses from the previous system. We apply a Japanese dependency parser KNP with JUMAN to make the clauses including a set of a predicate, a subject, and an object. When comparing a pair of sets between civil law articles and legal bar exams, we use a *precise match* and a *loose match*. The precise match performs exact matches of strings for its predicate, its subject and its object. The loose match compares either a pair of subjects or a pair of objects,

in addition to matching predicates. When our systems could not output any answer, our system answer *yes* as a default output. Additionally, when any negation appears in clauses, we reverse yes/no output.

3.2 Frame-Evoking Words

Our frame based system works like a part of semantic role labeling. Semantic Role Labeling (SRL) is a representative NLP task using FrameNet. The SRL has four processes: (i) identify a frame-evoking word, (ii) identify a frame from the frame-evoking word (frame disambiguation), (iii) estimate words, phrases, or clauses which we have to give FEs, (iv) labeling the FEs. Our frame based system corresponds to these (i) and (ii) processes.

To identify a frame-evoking word, we use a predicate in a proposition clause set which is given by our previous system. Next, we add a candidate of frame-evoking words from the predicate using Japanese WordNet to connect with a specific LU. This is because the number of frame evoking words contained in a LU is small.

We use either English LUs or Japanese LUs. When we use English LUs, we add English translation words in Japanese WordNet to the candidate of word-evoking words. When we use Japanese LUs, we add synonyms, hypernyms and hyponyms in Japanese WordNet to the candidate of word-evoking words. We select LUs which contain one of the word-evoking words. Finally, we make a candidate of frames from the selected LUs. Fig. 2 shows an example of this process.

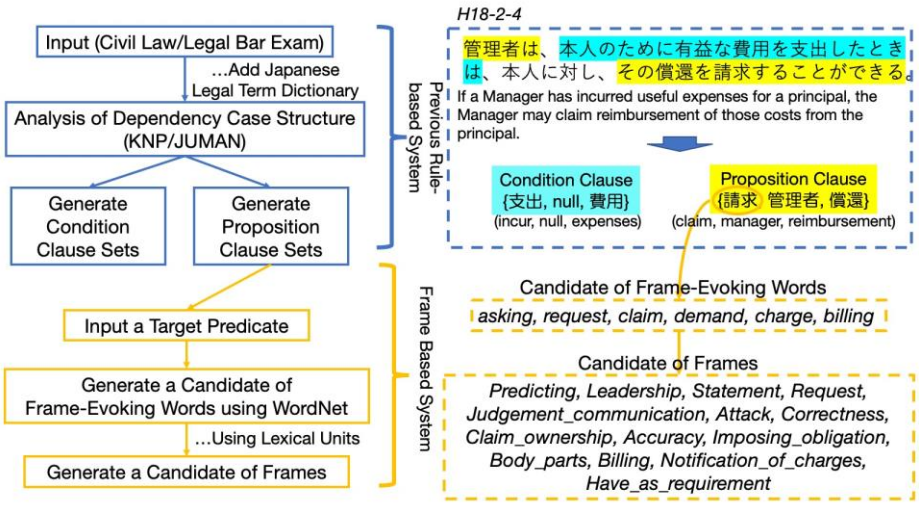


Fig. 2. An example of our frame detection process, using English LUs.

3.3 Frame Disambiguation and Metrics of Frame Confidence

We compare a pair of frame candidates in round robin. We take a pair of frames which confidence value is highest. To calculate the confidence between two frames, we use a shortest path determined by the Dijkstra Algorithm [39] from the entire graph of the frame relations. We assigned a weight value to all of the frame relation types (Table 1). These weights are determined by heuristics. When a weight value is higher between a pair of frames, then we regard this pair as more similar. The following four examples are some of the typical relations.

Inheritance is the strongest relation between frames, corresponding to is-a relationship. So, each frame element in a parent frame should correspond to a frame element in its child frame. Therefore, we set the highest value to this Frame Relation. *Using* is used in a part of a scene evoked by a child frame that refers to its parent frame; some parent frame elements might not have corresponding child frame elements. *Perspective_on* is similar to *Using*. While *Perspective_on* could treat at least two perspectivized frames (e.g. the Commercial_transaction frame specifies a complex scheme involving an exchange of subjects between a seller and a buyer). *Subframe* aggregates frames that form a complex sequence as a whole.

We define the confidence value as a multiplication by the weights of frame relations on the path (Fig. 3). We set threshold for confidence values. When a confidence value is beyond the threshold, we regard the corresponding pair of predicates as similar. Therefore, the lower threshold we set, the more pairs our system could compare. Then

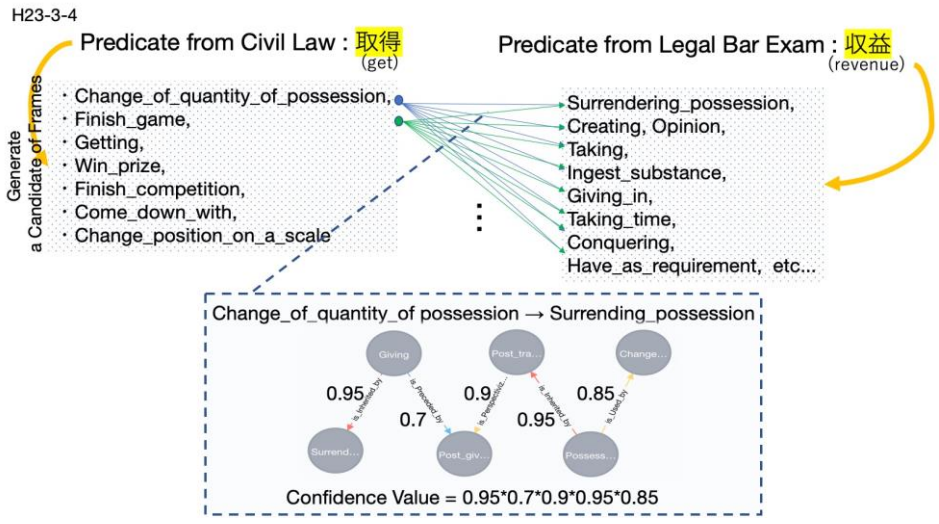


Fig. 3. An example of the confidence value, which is calculated by a multiplication by the weights of frame relations on the path.

we compare the corresponding clauses of civil law articles and legal bar exams extracted by our rule based system as same as our previous system, assuming the corresponding pair of predicates is identical.

FrameRelationType	RelevanceWeignt	FrameRelationType	RelevanceWeight
Inheritance	0.95	Inchoactive_of	0.65
Perspective_on	0.9	Causative_of	0.65
Using	0.85	Metaphor	0.5
Subframe	0.8	See_also	0.5
Precedes	0.7	ReFrameing_Mapping	0.5

Table 1. Weight values of the frame relation types.

4 Experiments and Results

Experiments were conducted on the COLIEE 2018 statute law competition data corpus (Task 4). We did not use the training data except for evaluations, because our frame based system does not use any machine learning method i.e. unsupervised.

4.1 COLIEE Datasets

In this paper, we focused on Task 4. Training data of Task 4's legal questions is drawn from the Japanese legal bar exams. Relevant Japanese civil law articles were also provided. While there was an English translation version of the dataset provided, we only used the original Japanese version. Fig. 4 shows an example of the COLIEE statue law competition data.

t1: （留置権の行使と債権の消滅時効）
第三百条　留置権の行使は、債権の消滅時効の進行を妨げない。
(Exercise of Rights of Retention and Extinctive Prescription of Claims)Article 300
The exercise of a right of retention shall not preclude the running of extinctive prescription of claims.
t2: 留置権者が留置物の占有を継続している間であっても，その被担保債権についての消滅時効は進行する。
Even while the holder of a right to retention continues the possession of the retained property, extinctive prescription runs for its secured claim.

Fig. 4. An example of COLIEE legal bar problem which asks to answer whether t1 entails t2 or not.

4.2 Performance Experiments

In order to investigate our frame based system performance, we used the COLIEE training dataset which includes the past legal bar exam problems and answers. We performed textual entailment part of Task 4, given the gold standard answer of Task 3. We compared a couple of combinations of our modules, in order to observe effects of the frame information. Because the COLIEE dataset is unbalanced, i.e. the number of yes answers and no answers are not equal. When our systems could not output any answers, we tried to fill with either all yes or all no answers as default output to normalize this unbalance. Table 2 shows the result of these performance experiments, using Japanese LUs.

Default Output	Score	PreciseMatch	PreciseMatch + FrameNet	LooseMatch	LooseMatch + FrameNet
N	Average	0.577	0.574	0.595	0.591
	CorrectNum	378	376	390	387
Y	Average	0.522	0.525	0.533	0.542
	CorrectNum	342	344	349	355

Table 2. Results of performance experiments. Our FrameNet system uses Japanese LUs with 0.9 as its threshold.

4.3 Formal Run Experiments

Table 3 shows our formal run results in COLIEE 2018 Task 4. The differences from the performance experiments above. The results of KIS_Frame based on the loose match, which uses English LUs and the threshold is 0.99. The number of comparable

Team	Language	# Correct Answers (total 69 answers)	Accuracy
YA	?	44	0.6388
KIS_Frame	J	39	0.5652
KIS_mo3	J	38	0.5507
KIS_dict	J	37	0.5362
KIS_SVM	J	36	0.5217
KIS_Frame2	J	35	0.5072
UE	E	33	0.4783

Table 3. The COLIEE 2018 formal run results. “J” is using Japanese test datasets, and “E” is English version.

pairs increases by using FrameNet, which becomes too many when comparing all of the civil law articles. We restrict the possible number of the comparisons by setting larger threshold in Task 4. We changed the threshold to 0.7 from 0.99 in the performance experiments, because we use the training datasets which include gold standard civil law articles to be compared with.

KIS_Frame2 is different from KIS_Frame in that the loose, rough and precise match modules are used together. In KIS_Frame2, we use English LUs and the threshold is 0.7.

5 Discussion

Firstly, we observed similar score distribution between the baselines and our frame based systems, among examination years from H18 to H28. Our system should have added new results, rather than changing the entire answer set drastically.

Secondly, there is almost no difference between the precise match results regardless of the FrameNet's effect. These results suggest that there was a little number of clauses in the training set, which the precise match is applicable i.e. a pair of triples (subject, object and predicate) matches. On the other hand, we observed differences in the loose matches. In order to analyze this effect in detail, we focus on H28 (H28-3-5) as shown in Fig. 5. Our previous system cannot compare predicates when their surface strings are different, even though they have comparable similar meanings. In contrast, our frame based system can handle such predicates of similar meanings even if their string forms differ. Our frame based system could answer more problems than our previous system for this reason, which is supported by the actual results.

Thirdly, our frame based system can recognize a pair of predicates which essentially shares a same or similar meaning. However, this is not always the cases. Predicates of abstract meanings, such as "do" and "become", tend to evoke more frames, which results in higher confidence values; used frames sometimes seem not related to the legal

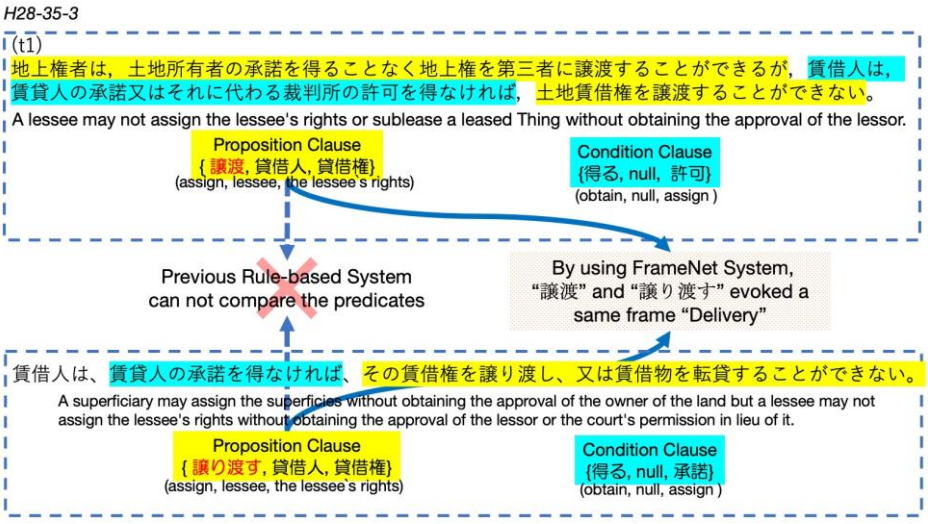


Fig. 5. An example of the effect of FrameNet.

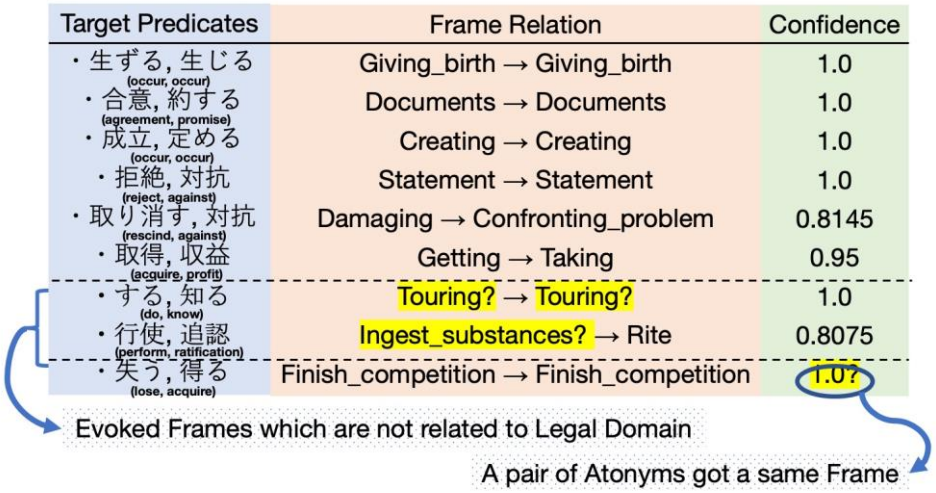


Fig. 6. An example of the analysis results.

domain; a pair of antonyms got the same frame, because they are typically used in the same situation. Fig. 6 shows examples of these cases.

Fourthly, whether using English LUs or Japanese one, we observe different evoked frames. For example, from the predicate "claim (請求)", our system acquired *Predicting*, *Leadership*, *Statement*, *Request*, *Judgment_communication*, *Attack*, *Correctness*, *Claim_ownership*, *Accuracy*, *Imposing_obligation*, *Body_parts*, *Billing*, *Notification_of_charges*, and *Have_as_requirement* Frames by using English LUs, in contrast *Claime_ownership* and *Have_as_requirement* by Japanese LUs. Precise analysis between English LUs and Japanese LUs would be needed to find an effects of FrameNet.

6 Future Work

Japanese text requires explicit tokenization process because there is no space between tokens. When this tokenization fails, final result could also be failed. Therefore, we need to refine the tokenization process and following predicate-argument structure analysis process to be optimized with our frame based system. For example, removing an abstract word could be effective.

We heuristically defined the weights of Frame Relations and the metrics of calculating the Frame confidence. Automatic tuning of the weights with some machine learning technique would be our future work. Using relevant graph theory could also improve the system.

The core of FrameNet are frame elements, in other words, semantical roles. By using frame elements, we could identify frames more precisely, capturing deeper semantic structures.

The most difficult issue to solve in legal domain would be the logic and abstraction, and how we approach these problems using FrameNet.

7 Conclusion

Legal document processing requires a variety of issues to be solved compared with other domains. The most distinguishing characteristic is that legal issues require precise analysis of a predicate argument structure and semantical abstraction. Based on this observation, we developed a yes/no question answering system for legal domain. Our system uses a Japanese case structure analyzer and FrameNet. We applied our system to COLIEE 2018 Japanese task (Task 4). Our system achieved the second best score among Task 4 participants, the best among our systems of different module combinations. We analyzed effectiveness of our frame based system by the training dataset, confirming increase of the scores when our frame based system was used.

Acknowledgements

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

References

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

[2] D. Ravichandran and E. Hovy, "Learning Surface Text Patterns for a Question Answering System," in *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, 2002, pp. 41–47.

[3] H. Yu and V. Hatzivassiloglou, "Towards answering opinion questions: separating facts from opinions and identifying the polarity of opinion sentences," in *Proceedings of the 2003 conference on Empirical methods in natural language processing*, 2003, pp. 129–136.

[4] D. Pinto, A. McCallum, X. Wei, and W. B. Croft, "Table extraction using conditional random fields," in *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, 2003, pp. 235–242.

[5] H. Cui, R. Sun, K. Li, M.-Y. Kan, and T.-S. Chua, "Question Answering Passage Retrieval Using Dependency Relations," in *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2005, pp. 400–407.

[6] X. Xue, J. Jeon, and W. B. Croft, "Retrieval models for question and answer archives," in *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, 2008, pp. 475–482.

[7] J. Bian, Y. Liu, E. Agichtein, and H. Zha, "Finding the Right Facts in the Crowd: Factoid Question Answering over Social Media," in *Proceedings of the 17th International Conference on World Wide Web*, 2008, pp. 467–476.

[8] E. M. Voorhees and D. K. Harman, *TREC: Experiment and Evaluation in Information Retrieval (Digital Libraries and Electronic Publishing)*. The MIT Press, 2005.

[9] N. Kando, K. Kuriyama, and T. Nozue, "NACSIS Test Collection Workshop (NTCIR-1) (Poster Abstract)," in *Proceedings of the 22Nd Annual International ACM SIGIR*

Conference on Research and Development in Information Retrieval, 1999, pp. 299–300.

[10] M. Braschler, “CLEF 2000 - Overview of Results,” in *Revised Papers from the Workshop of Cross-Language Evaluation Forum on Cross-Language Information Retrieval and Evaluation*, 2001, pp. 89–101.

[11] C. C. T. Kwok, O. Etzioni, and D. S. Weld, “Scaling Question Answering to the Web,” in *Proceedings of the 10th International Conference on World Wide Web*, 2001, pp. 150–161.

[12] O. Etzioni, M. Cafarella, D. Downey, S. Kok, A.-M. Popescu, T. Shaked, S. Soderland, D. S. Weld, and A. Yates, “Web-scale Information Extraction in Knowitall: (Preliminary Results),” in *Proceedings of the 13th International Conference on World Wide Web*, 2004, pp. 100–110.

[13] J. Jeon, W. B. Croft, and J. H. Lee, “Finding Similar Questions in Large Question and Answer Archives,” in *Proceedings of the 14th ACM International Conference on Information and Knowledge Management*, 2005, pp. 84–90.

[14] H. Kanayama, Y. Miyao, and J. Prager, “Answering Yes/No Questions via Question Inversion,” in *the 24th International Conference on Computational Linguistics (COLING 2012)*, 2012, pp. 1377–1391.

[15] S. Dumais, M. Banko, E. Brill, J. Lin, and A. Ng, “Web Question Answering: Is More Always Better?,” in *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2002, pp. 291–298.

[16] D. Ferrucci, “Introduction to ‘This is Watson,’” *IBM J. Res. Dev.*, vol. 56, no. 3.4, p. 1:1-1:15, 2012.

[17] D. A. Ferrucci, “IBM’s Watson/DeepQA,” *SIGARCH Comput. Arch. News*, vol. 39, no. 3, 2011.

[18] N. H. Arai, “The impact of AI—can a robot get into the University of Tokyo?,” *Natl. Sci. Rev.*, vol. 2, no. 2, pp. 135–136, Jun. 2015.

[19] “The Todai Robot project.” [Online]. Available: http://21robot.org/.

[20] I. Dagan, O. Glickman, and B. Magnini, “The PASCAL Recognising Textual Entailment Challenge,” in *Proceedings of the First International Conference on Machine Learning Challenges: Evaluating Predictive Uncertainty Visual Object Classification, and Recognizing Textual Entailment*, 2006, pp. 177–190.

[21] D. Giampiccolo, B. Magnini, I. Dagan, and B. Dolan, “The Third PASCAL Recognizing Textual Entailment Challenge,” in *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, 2007, pp. 1–9.

[22] M. Negri, A. Marchetti, Y. Mehdad, L. Bentivogli, and D. Giampiccolo, “Semeval-2012 Task 8: Cross-lingual Textual Entailment for Content Synchronization,” in *Proceedings of the First Joint Conference on Lexical and Computational Semantics - Volume 1: Proceedings of the Main Conference and the Shared Task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation*, 2012, pp. 399–407.

[23] H. Shima, H. Kanayama, C. Lee, C. Lin, T. Mitamura, Y. Miyao, S. Shi, and K. Takeda, “Overview of NTCIR-9 RITE: Recognizing Inference in TExt,” in *NTCIR-9 Workshop*, 2011, pp. 291–301.

[24] Y. Watanabe, Y. Miyao, J. Mizuno, T. Shibata, H. Kanayama, C.-W. Lee, C.-J. Lin, S. Shi, T. Mitamura, N. Kando, H. Shima, and K. Takeda, “Overview of the Recognizing Inference in Text (RITE-2) at NTCIR-10,” in *the NTCIR-10 Workshop*, 2013, pp. 385–404.

[25] S. Matsuyoshi, Y. Miyao, T. Shibata, C.-J. Lin, C.-W. Shih, Y. Watanabe, and T. Mitamura, “Overview of the NTCIR-11 Recognizing Inference in TExt and Validation (RITE-VAL) Task,” in *the 11th NTCIR (NII Testbeds and Community for information access Research) workshop*, 2014, pp. 223–232.

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

[27] M.-Y. Kim, R. Goebel, and S. Ken, “COLIEE-2015: Evaluation of Legal Question Answering” in *Ninth International Workshop on Juris-informatics (JURISIN 2015)*, 2015.

[28] M.-Y. Kim, R. Goebel, Y. Kano, and S. Ken, “COLIEE-2016: Evaluation of the Comptetion on Legal Information Extraction and Entailment” in *Tenth International Workshop on Juris-informatics (JURISIN 2016)*, 2016.

[29] Y. Kano, M.-Y. Kim, R. Goebel, and S. Ken, “Overview of COLIEE2017" in *International Conference on Artificial Intelligence and Law (ICAIL 2017)*, 2017.

[30] R. Taniguchi, and Y. Kano, “Legal Yes/No Question Answering System using Case-Role Analysis" in *Tenth International Workshop on Juris-informatics (JURISIN 2016)*, 2016.

[31] “JUMAN (a User-Extensible Morphological Analyzer for Japanese).” [Online]. Available: http://nlp.ist.i.kyoto-u.ac.jp/EN/index.php?JUMAN.

[32] “Japanese Dependency and Case Structure Analyzer KNP.” [Online]. Available: http://nlp.ist.i.kyoto-u.ac.jp/EN/index.php?KNP.

[33] Y. Kano, R. Hoshino, and R. Taniguchi, “Analyzable Legal Yes/No Question Answering System using Linguistic Structures " in *International Conference on Artificial Intelligence and Law (ICAIL 2017)*, 2017.

[34] “Welcome to FrameNet!.” [Online]. Available: https://framenet.icsi.berkeley.edu/fndrupal/.

[35] J. Ruppenhofer, M. Ellsworth, M. R. L Petruck, C. R. Johnson, C. F. Baker, J. Scheffczyk, “FrameNet II: Extended Theory and Practice”, 2016.

[36] C. J. Fillmore, “Frame Semantics”, in *Linguistics in the Morning Calm*, 1982, pp. 111–137.

[37] “Japanese FrameNet.” [Online]. Available: http://jfn.st.hc.keio.ac.jp.

[38] “Japanese WordNet.” [Online]. http://compling.hss.ntu.edu.sg/wnja/index.en.html.

[39] “Graph Compute with Neo4j: Built-in Algorithms, Spark & Extensions.” [Online]. https://neo4j.com/blog/graph-compute-neo4j-algorithms-spark-extensions/.

Question Answering System for Legal Bar Examination using Predicate Argument Structure

Reina Hoshino[1], Ryosuke Taniguchi[1], Naoki Kiyota[1] , and Yoshinobu Kano[1]

[1] Informatics Course, Graduate School of Integrated Science and Technology, Shizuoka University, 3-5-1 Johoku, Naka-ku, Hamamatsu City, Japan
rhoshino@kanolab.net, rtaniguchi@kanolab.net
nkiyota@kanolab.net, kano@inf.shizuoka.ac.jp

Abstract. We developed a question answering system for legal bar exam, which can explain the way system solves based on underlying logical structures. We focus on the set of subject and object with their predicate, i.e. the predicate argument structure, in order to represent structures of legal documents. We implemented a couple of modules using different searching methods. Our system outputs results using these modules by learning each module's confidence value with SVM. We manually analyzed the difficulty level of the problems whether external knowledge is required or not. We created a structured synonym dictionary specialized to the legal domain, where predicates are categorized with their objects. This synonym dictionary could absorb superficial differences of predicates to solve the problems which do not require external knowledge. We confirmed that the system can solve more than 70 percent of simple problems. Our system achieved the second best score in Task 4 of the COLIEE 2018 shared task.

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

1 Introduction

Automatic question answering for legal documents is gathering attention recently. This paper describes our challenge to Task 4 of the COLIEE 2018 legal bar exam, which asks participants to answer true or not based on the Civil Law Articles, given text drawn from the Japanese legal bar exam.

The COLIEE shared task series is held in association with the JURISIN (Jurisinformatics) workshop. The first one was the COLIEE 2014 shared task[1]. Following this, COLIEE 2015 shared task[2], COLIEE2016 shared task[3], and COLIEE 2017[4].shared task (this time in conjunction with ICAIL) were held. The COLIEE shared task consists of four tasks. We challenged Task four. Task one and Task two used the legal cases. Task one of this legal question answering task involves reading a legal bar exam question, and extracting a subset of Japanese Civil Code Articles. Task four requires both of the legal information retrieval system and textual entailment system. Given a set of legal yes/no questions, a participant's system will retrieve rele-

vant Civil Law articles. Then answer yes/no entailment relationship between input yes/no question and the retrieved articles. The corpus of legal questions is drawn from Japanese Legal Bar exams, and the relevant Japanese Civil Law articles were also provided. While there was an English translation version of the dataset provided, we only used the original Japanese version.

Regarding question answering in general, most successful systems employ rather traditional techniques of question answering which have decades of history [5]–[8]. Recently, automatic answering system used machine learning by big data. The typical example is IBM Watson System [9]. The core Watson system employed a couple of open source libraries, including the traditionally well-designed DeepQA system [10] as its skeleton of question answering processing.

However, there are only hundreds of problem sentences used in COLIEE. We think that the amount of data is not enough to perform an end-to-end supervised machine learning. There are an enormous number of legal documents, many methods have been proposed to acquire knowledge from case examples [11]. However, it is difficult to use case examples directly in the legal bar exam, because we need commonsense knowledge to follow the thinking of a legal expert, not just the explicit knowledge base available. In addition, a question answering system for legal documents is required to show reasons of its decision in a human interpretable way. Our aim is to create a legal question answering system where we can trace evidences of it decision.

Reasoning is essential in understanding a legislative system. Civil law articles consist of a legal effect part and cases to state its effect. A system needs to find them from problem sentences. However, there are a couple of difficult issues to find such relevant parts.

Firstly, the legal effect is not necessarily written in the article. We conducted an experiment asking Japanese native speaker to solve the legal bar exam while showing the related articles. These native speakers have never learned legal issues. Their accuracy was about 80 percent, which shows that people have little knowledge of law cannot solve all problems even the related articles are given. This means that external knowledge is important to solve the problems.

Secondly, there are ambiguities in legal articles. Words used in the law articles are different from the words in problem sentence in most cases, even when they express the same situation. For example, “成立する (effect)” and “適用する (apply)” are sometimes interchangeable. We created a synonym dictionary for predicates to hadle this issue. We made our synonym dictionary extracting from the past legal bar exam problems of six years (2009-2014). A predicate entry in our dictionary is structured to have a condition, what sort of object could be taken.

Thirdly, diversity of the vocabulary is a difficult issue. In the civil law, both legal technical terms and terms used in daily life are required, because civil law handles problems in daily life. We registered these words in our morphological analyzer’s dictionary..

The legal bar exam includes a variety of complex linguistic issues, such as synonym, commonsense knowledge, syntax, and semantics. to be resolved to understand legal documents. In order to make our system behavior interpretable, we focus on

simpler problems in this paper. We manually examined difficulty levels for tens of past years' problems.

Section 2 describes the proposed method of our question answering system. Section 3 shows results of COLIEE2018 and past years, including analyses of the difficulty levels. Section 4 concludes this paper, discussing future works.

2 System Architecture

2.1 System Design

Our question answering system consists of two parts: a related articles search part, and a question answering part. These parts use predicate argument structure analysis, comparing a pair of sentences based on case roles of the arguments. The article search part searches for the related articles by searching sentences of the same structure. The question answering part compares whether each sentence represents the same event. In addition, we prepare a couple of different modules with different judgment criteria, and make modules for each criterion. Our final answer is obtained by using SVM, which selects its output from the answers of the modules and their confidence values. These confidence values are calculated from the number of articles that our predicate argument matches with the problem sentences. The datasets we used were the civil law articles, the problems of the past legal bar exam.

2.2 Dictionary

We created a legal term dictionary extracted from "有斐閣法律用語辞典第4版" (Yuhikaku legal term dictionary fourth edition).

Table 1. Number of words used in civil bar exam

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

We examined vocabulary using past eleven years of problems, We found that more than 300 kinds of words are used in problems per one year. More than 50 new words appeared per year when registering words from older year to newer year (**Table 1**). We registered these words in our morphological analyzer's dictionary..

In addition to the dictionary above, we created an additional dictionary from morphological analysis results of the problems of 7 years. This dictionary makes our analysis of predicate argument structures correct. Predicate argument structure analysis often failed because morphological analysis was not correctly performed. This failure is due to unknown words which are not listed in our Japanese dictionary nor our legal term dictionary. We added 200 new words manually.

s We created our additional dictionary, extracting from source codes of PROLEG (Satoh, Asai, & Hurukawa, 2011). PROLEG is a logic programming language for the legal domain based on Prolog. They systemize internal structure of the law from the civil law articles and actual cases. The PROLEG system returns a case is legally valid or not, by giving arguments such as a defendant and an object. Function names and variable names of PROLEG are named by hand. These names are necessary when reading and understanding the legal documents deeply. Therefore, we could obtain composite words not listed in the normal legal term dictionary. We extracted function names and variable names that do not use alphabets nor numbers, then added them to our additional dictionary.

2.3 Predicate Argument Structure Analysis

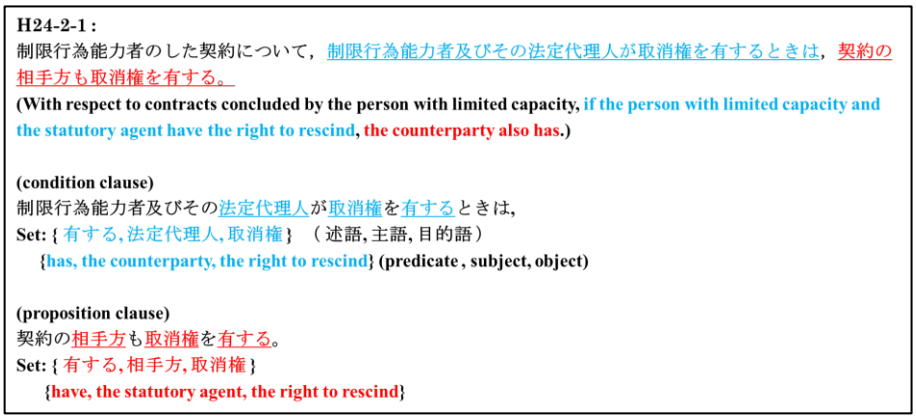


Fig. 1. An example of predicate Argument Structure Analysis

Fig. 1 shows an example of predicate argument structure analysis. We defined our own clause unit ("節") in order to recognize condition clauses and main clauses precisely, which are included in a single sentence. A clause should include a single predicate as a core element of that clause. We apply a dependency parser that makes chunks ("文節") of a couple of morphemes. Starting form a chunk that includes a

predicate, we aggregate neighboring chunks when a neighboring chunk does not include any predicate, until a clause unit is formed.

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

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

When searching related articles, we use a set of a predicate, a subject and an object as a predicate argument structure. For each sentence, we compare proposition clauses of the problem sentences and the civil law articles using these sets. The same applies for condition clauses. We use the base form of predicates for their comparison. For example, “認める (admit)” and “認めない (do not admit)” have the same meaning, sharing the same base form “認める (admit)”.

Normally, a sentence consists of a proposition clause and any number of condition clauses. Because an article consists of one or more sentences, we compare sentences one by one when comparing the article sentences with the problem sentence. However, when a problem consists of a couple of sentences, we regard the whole sentences as a single sentence. This is because the last sentence tends to be most important, while other sentences tend to represent conditions of the last sentence. For this reason, we regard clauses as condition clauses except for the proposition clause of the last sentence.

A difficulty in predicate argument structure analysis is an itemized article. An itemized article first describes the effect, then cases are listed to show the effect. In such a case, we cannot understand only with the first sentence. Sometimes there are no subjects or predicates in the part of case list. There are a lot of articles that have such a structure.

As a countermeasure, we connect the case list part with the first sentence. In this case, the first sentence always contains a phrase “次に掲げる (the following things)”. We delete this specific phrase and then insert the part of case list. We repeat this process for the number of items, making multiple sentences. However, the connected sentence could be syntactically invalid, causing errors in predicate argument structure analysis.

2.4 Synonym Dictionary for Predicates

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

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

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

Fig. 2. An example of synonym dictionary

We manually created a synonym dictionary for predicates. For example, “成立する (effect)” and “適用する (apply)” could have the same meaning. We used the problems of 2009 to 2006 to make our dictionary. We compared related articles and corresponding problem sentences to manually determine our synonym dictionary entries. As a result, we selected a combination of 33 verbs.

Our dictionary is structured with object conditions. For example, “果実を取得する (obtain fruits)” and “収益する (receive the profits)” have the same meaning when they have the same object.

We examined difficulty levels of the problems in order to verify how many problems can be solved by a synonym dictionary. The difficulty level is classified into four levels. The first level is “very easy”, corresponds to the problems which text is almost same with the article. The second level is “easy”, its problem can be solved by any Japanese native speakers without external knowledge. This “easy” problem requires synonym and/or zero pronouns resolutions. The third level is “difficult”, its problem requires general external knowledge but not legal expert knowledge. For example, a father’s father is a grandfather, which is a common sense of external knowledge. In addition, related articles in this level could span multiple sentences, sometimes become complicated structures. The fourth level is “very difficult”, its problem cannot be solved without external knowledge of the legal experts. In this level, its answer is not written in the related articles neither explicitly nor implicitly.

The “Target” column of **Table 2** shows the difficulty level statistics of the problems in 2014 and 2016. The “easy” problems share around 30 percent. We aim to solve these “easy” problems by our synonym dictionary.

Table 2. Difficulty level statistics of the past COLIEE problems

	very easy	easy	difficult	very difficult
2014	12	10	5	5
2016	21	32	30	10

2.5 Person Estimation

Some legal bar problems ask conceptual things, or ask how rights are oved using concrete stories. For example, a typical problem of concrete parable story describes persons called “A”, “B”, and “C”. In order to answer such concrete parable stories, we implemented a person estimation feature. In our person estimation feature, our system searches for words that are actors in the sentence, i.e. words with “人(human)” or “者(person)”, e.g. “代理人(agent)”. Predicate argument structure analysis sometimes fails to find a subject. When our system cannot find a subject, we insert a word as a subject that is found by this person estimation feature.

2.6 Question Answering Module

We created four types of question answering modules: precise match, loose match, rough match, and FrameNet modules.

Precise match module extracts civil law articles which proposition clauses match with a given proposition clause of problem sentences. For each predicate, we find a subject and an object by their case markers. We perform exact matches for the predicate, its subject and its object. When we could not find any subject nor any object, we skip that sentence. If a negative expression is detected only in one side, we reverse the corresponding Yes/No answer. Our system repeats these processes for the number of the clauses of the problem sentences. The precise match module has the highest percentage of correct answers among the four modules. However, this module cannot find answers for about 70 percent of problems.

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

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

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

Furthermore, we prepare the FrameNet module. FrameNet[12] is an English semantical lexical database based on a theory of meaning called frame semantics. The basic idea is that people understand the meaning of words largely by the frames which they evoke. Frames have some semantic roles called Frame Elements (FEs) and frame evoking words are called Lexical Units (LUs). We use LUs of Japanese FrameNet[13], in addition to the original English version of FrameNet.

We use Japanese WordNet[14], which is a lexical database for Japanese. We use Japanese WordNet to obtain synonyms, hypernyms, hyponyms and English translation words to expand FrameNet.

The FrameNet module is made by extending the loose match module. The difference of the FrameNet module from the loose match module is in comparing predicates. Using an inheritance relationship of FrameNet, we examine whether a pair of predicates represents the same meaning or not. This system measures a distance between nodes of words and regards as the same meaning if the nodes are in a near distance. We defined confidence values for each relationship type to calculate the distance.

In addition, we made more sub-modules by setting different options for these four modules. An option forces the predicate selection; when a clause includes specific condition pattern like "の場合は(in case of)" or "の時は(when)", a previous chunk is regarded as the core predicate of that clause. Another option is in the FrameNet module; we changed the confidence calculation method and/or the threshold of the confidence values.

2.7 Module Integration

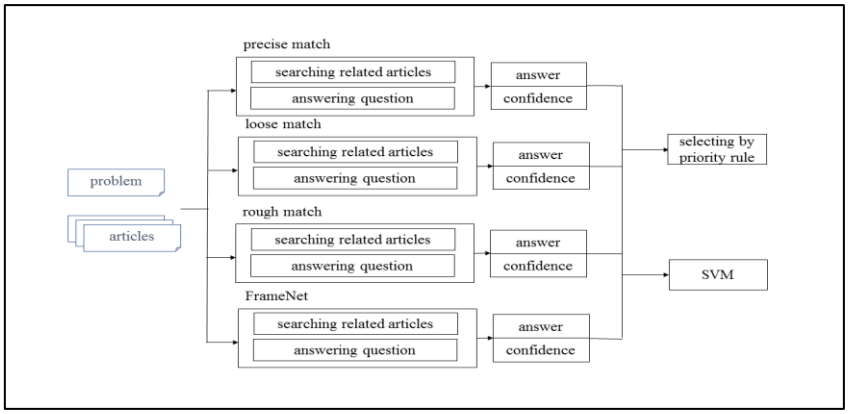


Fig. 3 Overview of our module integration

After completing the answers for each module, we decide our final answer. We prepared two ways for resolution.

When we do not use SVM, we try applying the precise match module first. When the precise match module cannot be applied, we apply the loose match module. If the

loose match module cannot be applied as well, we try applying the rough match module.

When we use SVM, we use each answer output by each module and the confidence calculated from the number of articles that matched word sets of the problem sentence for learning. The confidence is a value that decreases as more the number of the related articles. The equation of calculating the confidence is 1 / n. n is the number of the related articles when n > 0. The confidence is 0 when n = 0. However, the related articles of solving the problems are not only one, it is not necessarily true that the probability that the related articles is less and accurate is higher. The training data for SVM is the problems of 2006 to 2008.

The optimum cost and gamma value of SVM options were determined by grid search. The kernel used is a polynomial kernel. We used a kernel with a high rate of correct answers by comparing correct answer rates for each combination of functions and modules.

3 Experiment and Result

We applied our system to Task 4 of COLIEE 2018. In Task 4, only the problem sentences were provided without any additional information, asking Yes/No as answers.

3.1 Result of COLIEE 2018

Table 3. Result of COLIEE 2018 formal run

Team	Language	# Correct Answers (total 69 answers)	Accuracy
YA	?	44	0.6388
KIS_Frame	J	39	0.5652
KIS_mo3	J	38	0.5507
KIS_dict	J	37	0.5362
KIS_SVM	J	36	0.5217
KIS_Frame2	J	35	0.5072
UE	E	33	0.4783

Table 3 shows the result of the COLIEE 2018 Task 4 formal run. Among these results, team names with KIS as their prefix show our results. KIS_Frame uses the FrameNet module only. KIS_mo3 uses the precise match module, the loose match module and the rough match module. We selected options of these modules to be the highest correct answer rate on the training set. KIS_dict uses the same module as KIS_mo3 while adding our synonym dictionary. KIS_SVM uses SVM for mod-ule integration; other results integrates modules in the filter based way. KIS_Frame2 uses the FrameNet module, the precise match module, the loose match module, and the rough match module. KIS_Frame was the best result among our submissions for the COLIEE 2018 formal run.

There is little difference between the results of KIS_dict and KIS_mo3, synonym dictionary was used or not. Because many problems cannot be answered if the system do not handle synonyms, the effect of our synonym dictionary would have reduced due to other issues. As related articles are not given in Task 4, a huge number of combinations are compared by precise match and loose match. Our system does not have any feature to rank relevant articles. If there are many related articles, even if the correct judgment can be made by a synonym dictionary in the correct article, the result could be buried. Refining our searching meth-od for the related articles is the future work. The good result of KIS_Frame was probably because the system was able to effectively compare the predicates by the frame information.

3.2 Result of past years

Table 4. Result of COLIEE training data (past years's legal bar exam)

	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015	2016	total
KIS_SVM	0.54	0.57	0.60	0.66	0.51	0.65	0.53	0.62	0.60	0.52	0.53	0.57
KIS_mo3	0.57	0.57	0.60	0.60	0.48	0.63	0.51	0.63	0.57	0.52	0.55	0.56
KIS_dict	0.57	0.57	0.52	0.62	0.53	0.61	0.47	0.57	0.57	0.50	0.55	0.55
KIS_Frame	0.48	0.50	0.52	0.60	0.42	0.48	0.55	0.60	0.54	0.48	0.60	0.53
KIS_Frame2	0.51	0.59	0.62	0.58	0.46	0.53	0.44	0.44	0.53	0.50	0.42	0.51

Table 4 shows results of COLIEE training data of past years' legal bar exam. KIS_SVM, which uses SVM to mix different modules, performed best in total. However, KIS_SVM is ranked fourth in the formal run, lower than our other runs shown in the table. Its reason might be fluctuations between different year's problems. Especially, the problems of this year's formal run include many concrete stories. Moreover, this year's formal run includes problems consist of more than two sentences which were not appeared in the past problems. Because the tendency could have been different for these reasons, training model of SVM might not have fit well with the formal run data.

3.3 Comparison of Results for Difficulty Levels

Table 5. Results for each difficulty level

	Target	Accuracy
very easy	40	0.75
easy	62	0.387
difficult	44	0.522
very difficult	21	0.380

We manually categorized problems into four difficulty levels. shows results of the problem in 2014 and 2016. "Very easy" problems were correctly answered by more

than 70 percent, while “easy” problems and “difficult” problems were less than 40 percent. “Very easy” problems were solved well because our precise match can easily find superficially similar expressions in such problems. It was unexpected that “difficult” problems were solved better than “easy” problems. Even in problems that humans feel easy, synonym and zero pronouns resolutions are required, which are very difficult issues for computers to be solved. Regarding “difficult” problems, a very complicated structure could be transformed into a simple structure by predicate argument structure analysis, which is easier to solve than the “easy” problems. “Easy” and “very difficult” were far less than the random baseline (50%).

4 Conclusion and Future Work

We developed an automatic answering system for legal bar exam, which can show logical structures how the system solved a given legal problem, We focused on answering simpler problems in this paper. We confirmed that our system could correctly answer more than 70 percent of the simpler problems.

In order to represent structures of legal documents, we implemented a textual entailment solver, which has predicate argument structure analysis as its core part. We made a variation of different modules using different comparison methods. We used SVM to make ensemble of these modules, using confidence values calculated from the number of related articles.

In addition, we made a legal synonym dictionary. Our synonym dictionary is structured in that a s synonym of a predicate could be conditioned with its object. Unfortunately, we could not achieve a significant difference because there are too many related articles when searching relevant articles. We applied our system to the COLIEE 2018 Task 4 dataset. One of our submission achieved the second rank among participants.

The information retrieval part, which corresponds to Task 3, could be improved to increase the entire system performance in future..

Acknowledgements

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

References

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

[2] M.-Y. Kim, R. Goebel, and S. Ken, “COLIEE-2015: Evaluation of Legal Question Answering,” in *Ninth International Workshop on Juris-informatics (JURISIN 2015)*, 2015.

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

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

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

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

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

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

[9] D. A. Ferrucci, “Introduction to ‘This is Watson,’” *IBM J. Res. Dev.*, vol. 56, no. 3.4, p. 1:1-1:15, 2012.

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

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

[12] “Welcome to FrameNet!” [Online]. Available: https://framenet.icsi.berkeley.edu/fndrupal/.

[13] “Japanese FrameNet.” [Online]. Available: http://jfn.st.hc.keio.ac.jp.

[14] “Japanese WordNet.” [Online]. Available: http://compling.hss.ntu.edu.sg/wnja/index.en.html.

K NEAREST NEIGHBOR SEARCH APPROACH TO LEGAL PRECEDENCE RETRIEVAL

Moemedi Lefoane, Tshepho Koboyatshwene, and Lakshmi Narasimhan

University of Botswana
{moemedi.lefoane,tshepho.koboyatshwene,lakshmi.narashimhan}@mopipi.ub.bw

Abstract. People increasingly rely on digital technologies to store and access information they need. Among those are the precedence court case judgments which form part on the law in nations that adopt common law system. As such legal practitioners have a challenge of retrieving previous cases that can help support their arguments from the repositories of prior cases which grow exponentially every year. This make retrieval of prior cases a challenging task. Over the years research advancements in research communities such as Natural Language Processing, Information Retrieval, Information Extraction as well as Machine Learning provided solution to help effectively retrieve relevant cases. We propose Information Retrieval and Unsupervised Learning approach to the task of supporting cases for a given query. Supporting cases are also referred to as noticed cases: Specifically we deploy Information retrieval approach where *term frequency-inverse document frequency* is used as a similarity measure between a query and candidate cases, in addition we use k-nearest neighbor search still using *term frequency-inverse document frequency* as a distance measure for this task. In our approach we first rank documents according to their relevance to the query, we then apply filtering to exclude lowest scoring documents from relevant cases, using threshold value to cut off lowest scoring case judgments. Evaluation of our results show performance of 72% recall and 62% recall on information retrieval approach and unsupervised learning approach respectively. We however observe relatively lower scores when we compare precision performance scores to recall scores.

Keywords: legal Precedence Information Retrieval · Nearest Neighbor Search · TFIDF.

1 Introduction

A user (a legal practitioners working on a case) usually approach an information retrieval system with an information need. Information need is presented in the form of a query. An example of a query is a case judgment that a user is reading and the information need is finding similar cases that can help support arguments in cases they are working on. For this task a query is derived from fact

and summary of a case of interest. The information need is finding supporting cases from a collection of candidate cases, cited cases are likely to discuss the same issues as discussed in the query. Several approaches could be used help address the users' information need, they range from machine learning approaches, natural language processing, information retrieval, information extraction and other. where unsupervised machine learning approaches could be deployed such as k-means clustering, k-nearest neighbor search and others. Other solutions are Information Retrieval approaches with term weighting models such as BM25, *term frequency-inverse document frequency* (TFIDF) and others. Other research advancement include query expansion and formulation that aim to improve retrieval effectiveness by improving the query terms. Other approaches include natural language processing techniques that investigate the effect of parts of speech tagging and how they can be applied to improve retrieval effectiveness.

In our study we investigate the use of unsupervised learning approach to precedence retrieval, specifically we deploy K-Nearest neighbor search and TFIDF on noticed cases retrieval. Trstenjak et al [3] investigated the possibility of applying K-Nearest Neighbor search with TFIDF for text classification by finding nearest neighbors articles of different categories, they reported promising results with their approach. We follow the same approach by using K-Nearest neighbor algorithm with TFIDF, however we do not classify case judgments into different categories but we find k-nearest neighbors as first level ranking of candidate case judgments, we then apply filtering to help cut out less relevant candidate case judgments.

Section 1 provided a brief introduction. The rest of the paper is organised as follows; Section 2 outlines our proposed approach in details. Finally Section 3, 4, 5 and 6 discuss evaluation measures used, results, discussion and conclusion respectively.

2 Proposed Approach

2.1 Dataset

The dataset used was compiled by COLIEE2018 Case Law competition organizers. As outlined below the dataset was divided into 2; namely training and test set:

- Training Set:
  - 285 Facts and Summaries - These are used to derive queries.
  - 200 Candidate Cases for each of the 285 Queries - from these 200 candidate cases, models were developed to retrieve supporting cases from the candidate cases. The supporting cases (also referred to as noticed cases) were then compiled and sent to COLIEE2018 organisers for evaluation.
  - Ground truths for each of the 285 cases were provided, i.e. all the supporting cases for each of the 285 facts and summaries. Using ground truths models developed were evaluated and optimized.
- Test Set:

- 59 facts and summaries for the cases.
- 200 Candidate Cases for each of the 59 facts and summaries of cases(queries). Models developed using training set were then used on the training set to generate supporting cases runs. The runs were sent to COLIEE2018 organisers for evaluation.

2.2 Architecture of the System

Figure 1 shows the architecture of the proposed system. The system receives summary and fact of the case and it is passed through the query generator. Query generator derives the query from both fact and summary. The query generator uses TFIDF scores of terms in the fact, summary as well as the 200 candidate cases. Terms from fact and summary are combined and 50% of the lowest scoring terms are dropped. The remaining forms the query. The query is then used to retrieve supporting cases from candidate cases. Before retrieval process candidate cases are passed through the indexer which creates an index. Once the index is created, the retriever then use it to create a ranked list of 200 candidate cases matching the query. The ranked list has the highest scoring cases at the top of the ranked list. The ranked list is passed to the filter. The filter either gives top 25% of the 200 candidate cases as noticed for each of the queries. The filter can also apply the post filtering approach that computes the threshold score that is used to drop the lowest scoring cases. The result of the filter process is noticed cases for a given query. The noticed cases produced can be evaluated at this point to measure retrieval effectiveness. Further details on how the experiment were performed are provided on Section 2.3

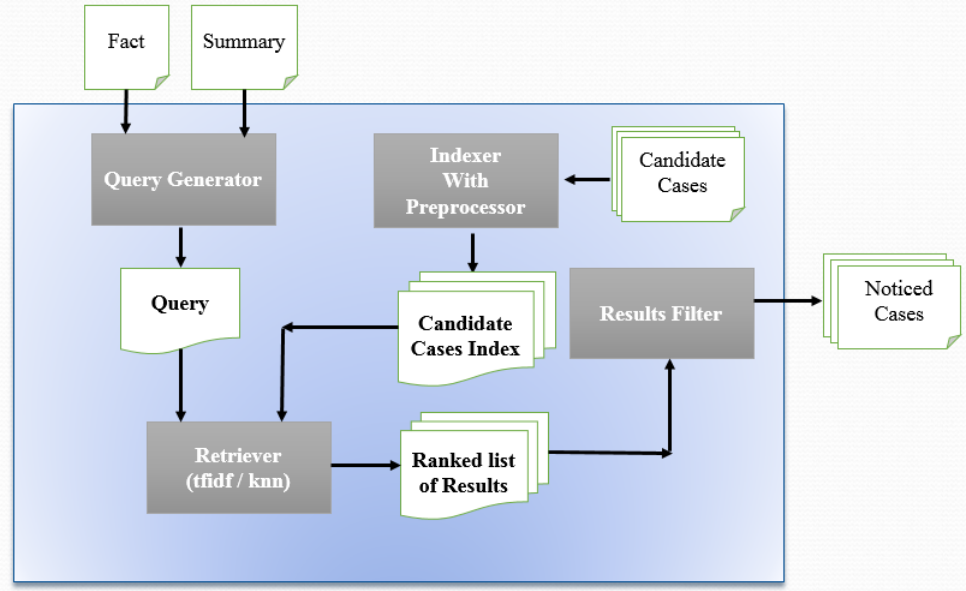
2.3 Experimental Setup

We submitted 3 runs for Task 1, noticed cases retrieval. For all the 3 runs we combined both fact and summary of a case, then computed TF_IDF of both fact and summary, we then drop 50% of the lowest scoring terms, the resulting terms we used as a query for our experiments, the intuition here was to try and keep the most informative term of the query in an attempt to avoid drifting of the results.

For our experiments we used Terrier 4.2 [4], Terrier is a widely used Information Retrieval open source research platform for indexing, retrieval and evaluation. It is a platform that has been used successfully for ad-hoc retrieval tasks, it is for this reason that we chose it for this task. For preprocessing steps we performed stopword removal, we did not do stemming on the candidate cases. For K-Nearest neighbor search we used graphlab create.

The first run was generated by deploying K-Nearest neighbor search algorithm with TFIDF and ranking candidates cases for each of the candidate cases. we then eliminated 75% of the lowest scoring candidate cases and the remaining formed noticed cases. The second run was generated by TFIDF which also produced a ranked list of noticed cases according to the degree of relevance, we then applied our filtering approach which compute the threshold scores for the

Fig. 1. System Architecture



candidate cases to eliminate. For the filtering approach we computed an average score of top 5 documents closest to the query then we divided the average by 2, the result we used as a threshold score that we used to eliminate any documents that scored lower than the threshold. The final run was generated in the same way as run one, but instead of eliminating 75% of the lowest scoring candidate cases we applied our filtering approach used in run 2.

For run 2 we used traditional IR approach and we choose the TFIDF as the document ranking model, we chose this model as it captures the relevance of document by matching query terms with candidate case documents with emphasis on rare terms.

3 Evaluation

Test collection based evaluation as outlined by Sanderson et al [1] dates back to 1950s, it has been adopted widely in research particularly in information retrieval, even to date it seems it is still the preferred evaluation measure. In their Book Manning et al [2] explain the measures of interest in evaluation for Information Retrieval. Section 3.1- 3.2 explain *Recall* and *Precision* in more details.

3.1 Recall

Recall is defined by Manning et al [2] as the proportion of the relevant documents that are retrieved, given by the formula in Figure 2, where *tp* is true

positive and fn is false negative. For noticed cases retrieval, tp refers to candidate cases that are correctly classified noticed cases retrieved by the system. On the other hand fn refers to candidate cases that should be retrieved as noticed cases but are not retrieved by the system.

$$Recall = \frac{tp}{tp + fn}$$

Fig. 2. Recall equation as described by Manning et al [2]

3.2 Precision

Manning et al [2] defines further $Precision$ as the proportion of the retrieved documents that are relevant. The formula for Precision is given in Figure 3, where tp is true positive and fp is false positive. For this task tp is as defined in Section 3.1. fp refers to candidate cases that are not noticed cases but are retrieved by the system as noticed cases.

$$Precision = \frac{tp}{tp + fp}$$

Fig. 3. Precision equation as described by Manning et al [2]

4 Results

Table 1 shows results of the runs submitted for evaluation by our team. As explained in Section 2.3 the first run was generated using K-Nearest Neighbor search, the second run was generated using TFIDF and the final run was generated using K-Nearest Neighbor search with post filtering approach. Table 2 on the other hand shows results for other COLIEE2018 task 1 participants. The results shown in the two tables are overall performance results for all queries. These results do not provide detailed information regarding individual query performance. Smith M.R et al [5] points out the importance of performing instance level analysis, it can help identify queries that are not classified correctly. Instance level analysis could also help explain why the queries are not classified correctly, ultimately explaining the overall performance failure. Furthermore information on individual query performance is vital and could lead to better conclusion and help inform direction for further research. As such instance level

analysis was performed to investigate the retrieval effectiveness further, the results are shown in Figure 4 to 7. For TFIDF approach no runs were submitted for evaluation as only 3 runs could be submitted by each team. In order to compare performance for K-Nearest neighbor search model and TFIDF IR system under the same conditions, TFIDF system approach was used to generate 25% of Top ranking candidate cases. It was then evaluated and used to compare the two systems. Figure 4 shows a comparison of precision performance of 2 TFIDF runs; one generated by giving 25% of top ranking candidate cases as noticed cases, and the second run generated by deploying post filtering approach. By applying post filtering approach precision consistently increase for a number of queries, but not for all. Figure 5 on the other hand compare recall performance from runs also generated by TFIDF IR approach. Consistently recall does not change for a number of queries when post filtering approach is applied. but for some queries post filtering approach cut of some of the supporting cases, leading to lower recall. Figure 6 and figure 7 shows instance level analysis for runs generated by K-Nearest neighbor search model. It reveals that post filtering approach for some queries cuts out most of the cases up to the top of the ranked list. This lead to a lower recall score for most of the queries. However for some of the queries precision does increase. These results are discussed in further in section 5 and conclusion is drawn.

Table 1. Results for the runs submitted.

Participant ID	Model	Precision	Recall	F-measure
UBIRLED-1	KNN-25%Candidate as noticed cases	0.132881	0.623211	0.219056
UBIRLED-2	TFIDF-post filtering applied	0.195511	0.720191	0.307536
UBIRLED-3	KNN-post filtering applied	0.561404	0.101749	0.172275

Table 2. Results for Other participants' the runs submitted.

Participant ID	Precision	Recall	F-measure
HUKB1	0.497436	0.308426	0.380765
HUKB2	0.404661	0.303657	0.346957
JNLP-r=2.5	0.546419	0.655008	0.595806
JNLP-k=10	0.676271	0.634340	0.654635
Smartlaw	0.287076	0.430843	0.344565
UA	0.372477	0.322734	0.345826
UA-postproc	0.348422	0.403816	0.374080
UA-smote	0.353868	0.392687	0.372268
UL	0.563798	0.302067	0.393375

Fig. 4. Instance Level Precision Performance for TFIDF approach

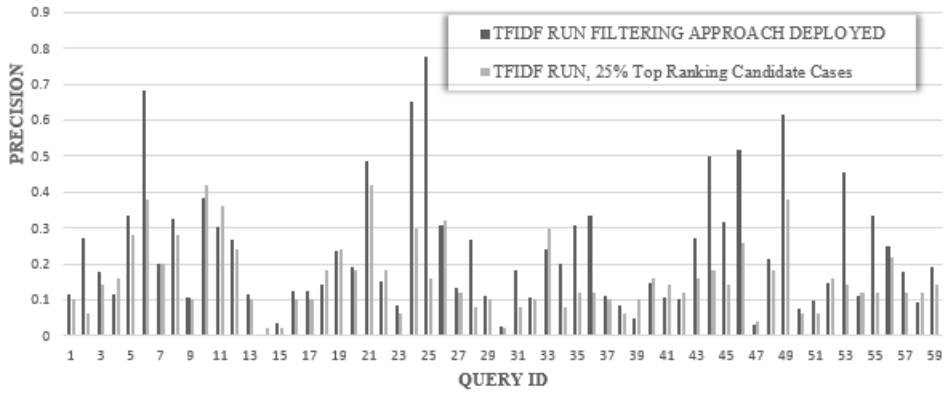
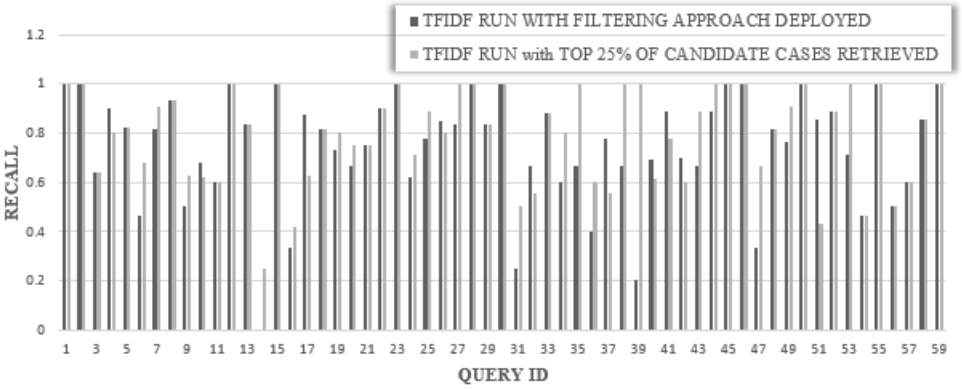


Fig. 5. Instance Level Recall Performance for TFIDF approach



5 Discussion

Results from table 1 shows that IR approach performs better than K-Nearest neighbor search, both in terms of precision and recall. In comparison with other participants' systems shown in Table 2, our system retrieve relevant cases with high recall comparable to other participants. While recall scores are high low

Fig. 6. Instance Level Precision Performance for K-Nearest Neighbor search

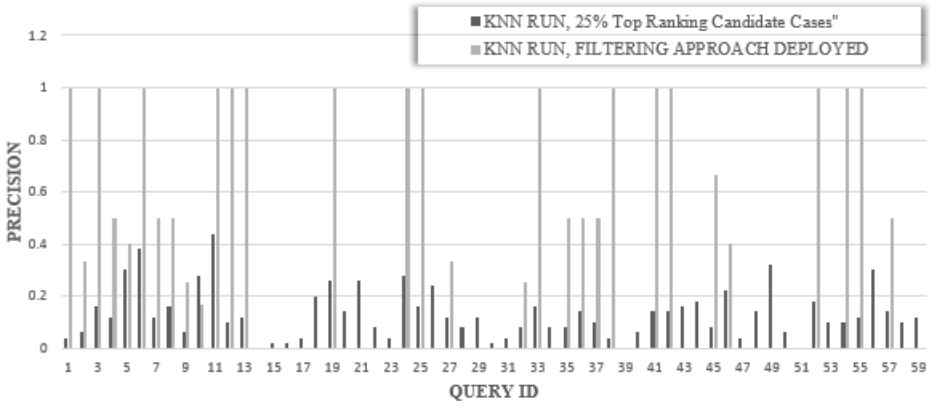
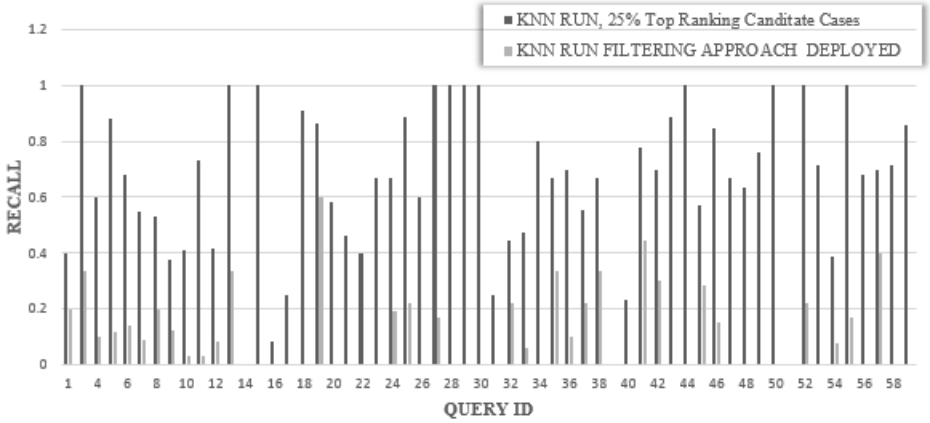


Fig. 7. Instance Level Recall Performance for K-Nearest Neighbor search



scores in precision are observed. This is suggest that for some queries our system retrieve considerably more cases that do not support a given summary and fact than supporting cases. For K-Nearest Neighbor search, employing post filtering approach to the ranked list generated (UBIRLED-1 and UBIRLED-3) an increase in precision is observed. On the other recall score for K-Nearest Neighbor search drops considerably when filtering approach is deployed. The increase in precision does not mean good performance if recall is not taken into consideration. The low recall score suggest that while post filtering cut cases that are not supporting a given query, it also cut out more supporting cases. For the information retrieval approach where we used TFIDF term weighting model, we evaluate a run that retrieves 25% of the candidate cases as noticed cases in order to compare IR approach to Unsupervised learning approach under the same condition. Instance level analysis was then perform to investigate the performance

for individual queries. Instance level analysis reveals that post filtering approach employed consistently increase precision. On the other hand for a number of cases recall remains the same even after post filtering is deployed on the ranked list. For other queries it is observed that filtering approach does cut out some of the noticed cases, leading to reduced recall values.

6 Conclusion

This paper set out to investigate retrieval effectiveness of applying unsupervised learning approach to retrieval of supporting cases to given facts and summary of a case. Information retrieval was used to compare and find out which of the two approaches performs better on this task. Results of our experiments shows slightly better performance on applying information approaches over unsupervised. However the difference in performance is not sufficient enough to conclude without further studies as reveled by instance level analysis. Our post filtering approach learned heuristically, works for some of the queries but not all of them. As future work boosting and Bragging Techniques should be investigated in order to find out if by giving higher weight to documents with certain features and penalizing those without such features. The candidate cases have some structure, for instance, headnote, summary and citations. With this structure Field based retrieval techniques should be investigated to exploit this structure. Other techniques include query difficulty predictors.

References

1. Sanderson, M.: Test Collection Based Evaluation of Information Retrieval Systems. Foundations and Trends in Information Retrieval, vol. 4, 247–375 (2010)https://doi.org/10.1561/1500000009
2. Manning, C. D., Raghavan, P., Schtze, H.: Introduction to Information Retrieval. Cambridge University Press, Cambridge, UK (2008)
3. Trstenjak, B., Mikac, S, Donko, D.: KNN with TF-IDF Based Framework for Text Categorization. 24th DAAAM International Symposium on Intelligent Manufacturing and Automation, 2013, Procedia Engineering, vol. 69, pp. 1356–1364. ScienceDirect. https://doi.org/10.1016/jproeng.2014.03.129
4. Ounis, I., Amati, G., Plachouras, V., He, B., Macdonald, C., Lioma, C.: Terrier: A High Performance and Scalable Information Retrieval Platform. Proceedings of ACM SIGIR'06 Workshop on Open Source Information Retrieval (OSIR 2006), Seattle, Washington, USA (2006). https://doi.org/10.10007/1234567890
5. Smith, M.R.,Martinez, T.,Giraud-Carrier, C.:An Instance Level Analysis of Data Complexity. Machine Learning (2014), vol. 95, pp. 225–256. Springer.

Author Index

ISBN 978-4-915905-82-7 C3004(JSAI)