IOP Conf. Series: Journal of Physics: Conf. Series **887** (2017) 012057 doi:10.1088/1742-6596/887/1/012057

An ontology based trust verification of software license agreement

Wenhuan Lu[1, a], Xiaoqing Li[1, b], Zengqin Gan[1, c] and Jianguo Wei[1, d]

[1]School of Computer Software, Tianjin University, Tianjin 300350, China
Email: [a]wenhuan@tju.edu.cn, [b]xiaoqingli@tju.edu.cn, [c]zqg@tju.edu.cn, [d]jianguo@tju.edu.cn

Abstract. When we install software or download software, there will show up so big mass document to state the rights and obligations, for which lots of person are not patient to read it or understand it. That would may make users feel distrust for the software. In this paper, we propose an ontology based verification for Software License Agreement. First of all, this work proposed an ontology model for domain of Software License Agreement. The domain ontology is constructed by proposed methodology according to copyright laws and 30 software license agreements. The License Ontology can act as a part of generalized copyright law knowledge model, and also can work as visualization of software licenses. Based on this proposed ontology, a software license oriented text summarization approach is proposed which performances showing that it can improve the accuracy of software licenses summarizing. Based on the summarization, the underline purpose of the software license can be explicitly explored for trust verification.

1. Introduction

Once people install or download software, there is definitely showed up a Software License Agreement, which textually varies from license to license. Generally the agreement would be a long document with millions of words that is hard to be understand, even make users don't take time to read it before clicking agree button to install or download it. This situation makes users feel unsafe and uncomfortable when installing unfamiliar software.

Doesn't matter the different representations, forms and statements of the agreements, the intention of most software license agreements would be similar even be partially same. The semantic behind the literature of license agreements are same for lots of cases. As ontology is a leading edge technology for knowledge representation, we use conceptual modeling approach to build generalized software license agreement ontology, so as to figure out the common knowledge behind each license arguments [1, 2]. The ontology based knowledge model can be a shared representation for license agreement domain.

Software license agreement normally designed according to Copy-right Law [3]. The construction of software license agreement domain ontology is based on a large number of law articles and different software license agreements collected from internet. These sample data sets were taken from the official website of some popular software in computer science. It is important to analyze and summarize the concepts including their relations and conditions in this domain. Redundancy of concepts and properties can result the failure of the ontology construction.

The modeling of software license agreements is not a trivial issue and hardly to solve, especially concerning the diverse software and terminologies. Our research objective is to propose an appropriate methodology and a uniform conceptual model to provide a descriptive and strong expansibility

Content from this work may be used under the terms of the Creative Commons Attribution 3.0 licence. Any further distribution of this work must maintain attribution to the author(s) and the title of the work, journal citation and DOI.
Published under licence by IOP Publishing Ltd

semantic knowledge representation for the domain of software license agreement. Besides, the ontology model can also provide the visualization of the semantic of licenses in order to facilitating knowledge representation

A License Ontology plays an important role in knowledge acquisition of software license agreement. In this paper, we combine our License Ontology with existing text summarization technology [4]. The goal of our work is to take an information source, extract content from it and present the most important content to the user in a condensed form that is manner sensitive to the user's or applications' needs, and improve the algorithm of Classifier4J summarizer based on the features of the domain of software license agreement [5, 6]. With the help of License Ontology, we want to provide every software end users a utility recommendation system, as shown in figure 1, to acquire the most important part of software license agreement as soon as possible.

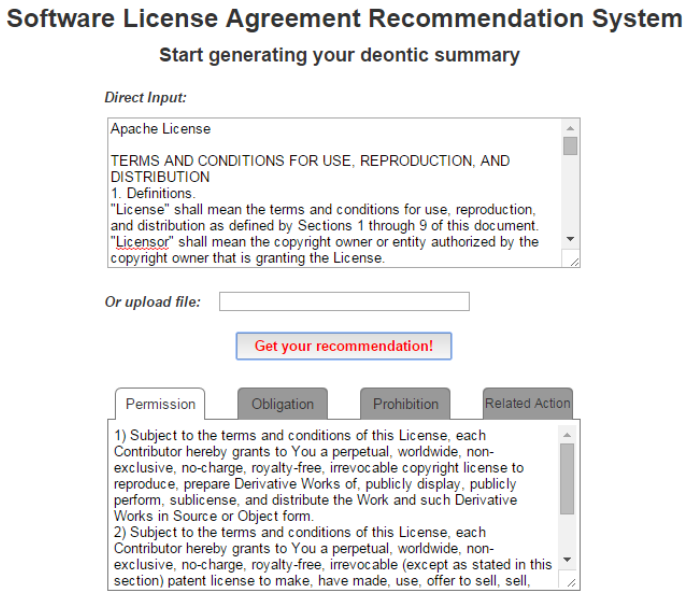

Figure 1. Prototype of Software License Agreement Recommendation System.

The interface is just a prototype so far and it is designed by Axure RP. All you need to do is upload a software license agreement and click the button and you will acquire the important part of the software license agreement as soon as possible.

The rest of this paper is organized as follows. First, Section 2 explores related work of the ontology construction. After that, methodology and model of domain ontology have been elaborated in Section 3. Then Section 0 presents our ontology for license case. Section 0 describes the experiment of software license agreement ontology. It combines ontology theory and text summarization and draws a conclusion by the result. Finally, the paper closes with concluding remarks and future perspectives.

2. Related Work

The methodology and model of domain ontology determine the describing and reasoning ability of ontology towards domain knowledge. A better model is the assurance of ontology construction.

As for the copyright domain which software license agreement belongs to, important efforts in this field have appeared in the last several decades overseas? However, related research is still at seed-time interiorly.

McCarthy introduced ontology into artificial intelligence for the first time in 1980. He proposed that we should construct an ontology to describe a world based on logical concepts [7]. The most frequently cited definition is that given by Gruber in 1993, that is, an ontology is defined as “an explicit specification of a conceptualization”. In 1998, Studer et al. modified it stating that: “Ontology is a formal, explicit specification of a shared conceptualization. [8]” Valente’s FOLaw distinguishes the various types of knowledge in legal reasoning, including normative knowledge, meta-legal

knowledge, world knowledge, responsibility knowledge, reactive knowledge, and creative knowledge[9]. Breuker's LRI-Core, a core ontology for law, from the thought of a common sense foundational ontology, including five major categories in the top layer of LRI-Core: physical, mental, abstract, role, and occurrence [1[10]]. On the basis of LRI-Core, Rinke Hoekstra's LKIF core legal ontology, which means Legal Knowledge Interchange Format, is a library of ontologies relevant to the legal domain [11, 12]. It consists of 15 modules, each of which describes a set of closely related concepts from both legal and commonsense domains.

These works try to formalize and systematize legal theory or knowledge, and provide a uniform and standard cognition for legal domain understanding and concepts spreading. However, there is no ontology specially oriented to software license agreement documents, for which general legal article could not work properly. One reason is that the statements in software license differ with the legal articles. Secondly, the legal articles are more formal structured than software license agreements in normal. The model need refer to both copyright ontology and software license agreement domain knowledge.

3. An Ontology Model of Software License Agreement

The software license agreement domain is literature rich domain, and the same concept could have many different statements. We conceptualize this domain in three phases that allow facing this process in an incremental way. Starting from the Role Model, we combine entities in the domain of software license agreement with actions, and build a core framework of the domain ontology. Secondly, there is the model for the permissions, prohibitions, and obligations part, the Deontic Model. A model for the available actions, the Action Model, is built on top of the two previous models, and then we construct an ontology.

3.1. Role Model

Before the construction of ontology, we divided Software License Agreement Role into Legal Entity Role and Action Role. A role is not a character role in the usual sense, but rather an entity that is played by another entity in a context.

Each role has its context, shown in figure 2 by "context", we mean something as a whole, including a relation in which the former "entity" is defined. The context of a Legal Entity Role is software license agreement, which means all of the concepts involved in Legal Entity Role are from software license agreements and related legal regulations. By "Legal Entity Role", we mean role holders who are played by contributors or users. They have actions which are related actions in the legal field and impact on the productions. The context of "Action Role" is deontic model which will be introduced next. In other words, the actions involved are mainly from deontic model, and they are played by Legal Entity Role. That is the integrated framework of software license agreement domain ontology.

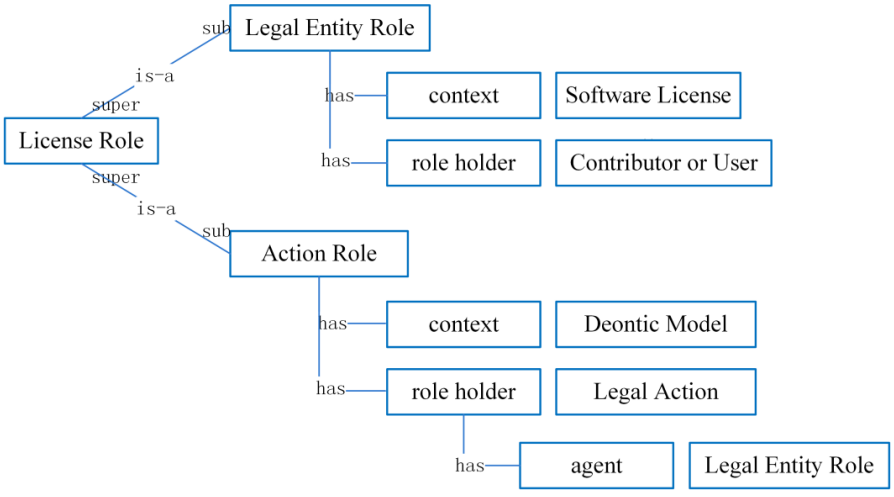

Figure 2. Role Model of software license agreement.

3.2. Deontic Model

After role modeling, we must build Deontic Model to integrate Role Model with software license agreement. It is important to note that legal regulations in this paper are from the World Intellectual Property Organization.

Legal regulations are similar in different software license agreement which can be divided into three parts. A permission allows an agent to perform the associated actions. A prohibition prevents an agent to perform the associated actions. An agent must perform an obligation which is the responsibility of the agent. The three parts contain most of the action in the domain of software license agreement.

A deontic act is performed by an agent under a certain condition and leads to a certain consequence. Sometimes the agent is obliged to do the deontic acts. We describe a deontic act as a tuple deontic (agent, action, condition, obligedTo, consequence).

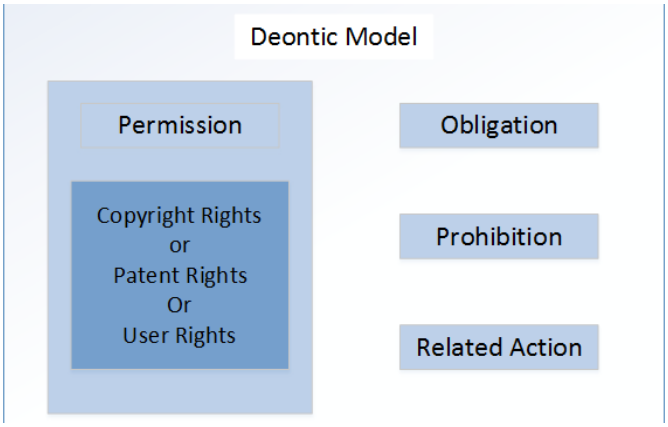

Figure 3. Deontic Model of software license agreement.

Figure 3 shows that deontic model contains permission, obligation, prohibition, and other related actions. Whether permission, prohibition, or obligation are describing the actions of the "agent". "Action" is the actions included in deontic model. "Conditions" signifies the precondition that an agent is allowed to execute the actions in usual. "ObligedTo" is the one who forces the agent to do the action. When you unfinished obligation or go against prohibition, "consequence" happen.

The most important part in deontic model is permissions of legal entity in the domain of software license agreement. Among all these grant licenses, related regulations of copyright rights and patent rights are from contributors, however, user rights are not from contributors, but from the law. In addition, related actions will be mentioned later.

3.3. Action Model

The last model, the Action Model, corresponds to the primitive actions that can be performed by the concepts defined in the Role Model and Deontic Model. These concepts are associated with a series of object properties, so that the ontology can illustrate what regulations expressing. Action model associate verbs primarily which is the main part of the model and the core of the software license agreement domain ontology.

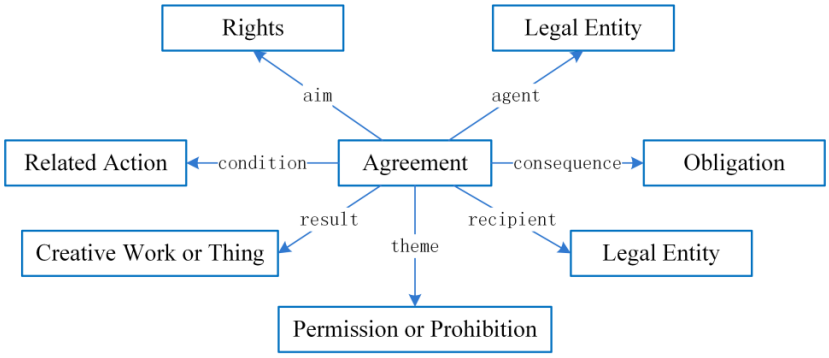

Figure 4. Action Model of software license agreement.

As it is shown in figure 4, there are main object properties related to actions:

- Agent: to execute, its domain is a legal entity.
- Aim: purpose of the action, its domain is rights.
- Consequence: result of the action, generally shows an action will certainly happen because of the other, its domain is obligation.
- Recipient: to accept, its domain is a legal entity.
- Result: product of the action, an inevitable product after legal entity is executed by the action.
- Condition: inverse of consequence, its domain is related action.
- Theme: object of the action, its domain is permission or prohibition.

The properties in our ontology are far more than these object properties shown in figure 4. Object properties mentioned above are basic properties which normalize the logical relationship of domain ontology, and there are some other properties, see Table 1, which play an important role during the construction of domain ontology using protégé 4.3.

Table 1. Object Properties of Action Model.

property	object	definition
agent	Legal Entity	voluntary initiator
aim	Rights	voluntary goal
consequence	Obligation	involuntary goal
recipient	Legal Entity	animate goal
result	Creative Work or Thing	inanimate goal
condition	Related Action	necessary circumstance
theme	Permission or Prohibition	essential participant
start	Date	dateminant temporal source
duration	Time	atemporal process resource
medium	Thing	physical resource for transmitting information
location	Place	spatial essential partipant

4. Instantiation of License Ontology

Ontology is the philosophical study of the nature of being, becoming, existence, or reality, as well as the basic categories of being and their relations. In 1993, Gruber originally defined the notion of ontology as an “explicit specification of conceptualization”. In 1998, Studer et al. modified it stating that: “Ontology is a formal, explicit specification of a shared conceptualization.” In recent years, after introducing into computer science, ontologies define domain concepts and the relationships between them, and thus providing a domain language is meaningful to both humans and machines. Ontologies have been shown to have benefits in a number of areas:

- knowledge sharing;
- knowledge reuse;

- verification and validation;
- domain theory development;
- Knowledge acquisition.

The previous pool of primitive legal entity and action roles, together with deontic properties that relate each action to the relevant roles, allows modeling the top-level of the software license agreement domain ontology, however, specific licenses should be associated with the top model to build the whole domain ontology. To reach the aforementioned research objective, we need to design and develop an ontology that conceptualizes different software license agreements from a uniform perspective. We use Protégé for ontology development of concepts and relationships. Protégé is a free, open source ontology editor developed at Stanford University. It provides a graphic user interface to define ontologies and deductive classifiers to validate the consistency of models and to infer new information based on the analysis of ontology. Recently, protégé has become the leading ontological engineering tool.

For instance, figure 5 shows how a software development product involves in the three top models between developing and using.

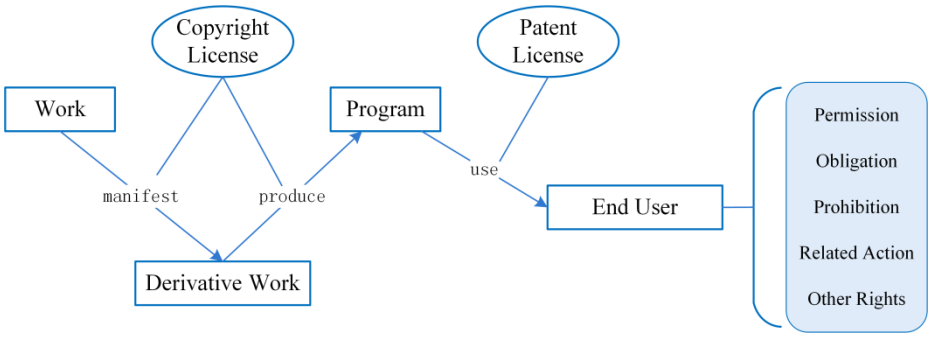

Figure 5. Software License Agreement Value Chain.

A work is realized as a manifestation after manufacture or other means that is used by end users under the software license agreement. In fact, every process can be regarded as an action model.

For example, Eclipse is open-source integrated development environment based on Java. By itself, it is simply a framework and a set of services for building a development environment from plug-in components. The version of Eclipse has been updated with the development and increasing demand, and after every update, there is a new derivative work. The compilation tool we use can be regarded as a product, and while the end user is using the product, it involves permission, prohibition, obligation, and other related actions stated in legal regulation as we mentioned before.

In the software license agreement field, there are numerous open-source software and business software, which have the very big different in regulation expression and the use of legal terms. Constructing domain ontology can formalize these software license agreements, and give an accurate description. Some classical legal ontologies that currently exist are wildly in legal field and clearly describe different object and the relationship between them. However, as an important branch of copyright, the domain of software license agreement needs a more applicable ontology. The ontology for software license agreement proposed in this paper is aimed at solving this problem.

In order to understand domain ontology better, event patterns should be introduced to state what is permitted, prohibited or obliged by a license and be naturally captured by the ontology terms described. The proposed actions and roles are used to model event patterns in the software license agreement.

4.1. Instantiation of Permission Pattern

For instance, figure 6 shows the sixth paragraph of article 10 of the Copyright Law of the People's Republic of China: The right of distribution, that is, the right to provide the public with original copies

or reproduced copies of works by means of selling or donating. Licensor has the right to distribute software permitted by WIPO and charge for the copies from end user.

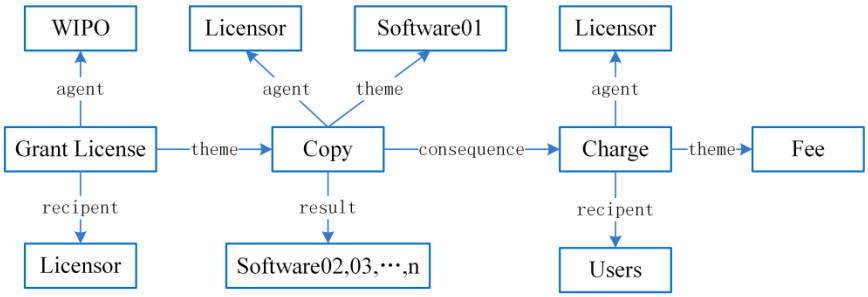

Figure 6. Instantiation of a Permission.

4.2. Instantiation of Obligation Pattern

Obligations are captured as event patterns that must be satisfied after the event pattern that triggers the obligation is exercised. For instance, figure 7 shows the legal regulation of Eclipse Public License v1.0: Under this section, the Commercial Contributor would have to defend claims against the other Contributors related to those performance claims and warranties, and if a court requires any other Contributor to pay damages as a result, the Commercial Contributor must pay those damages. That is, contributor has to compensate if there is any damage during use of the product, and meanwhile, called Commercial Contributor.

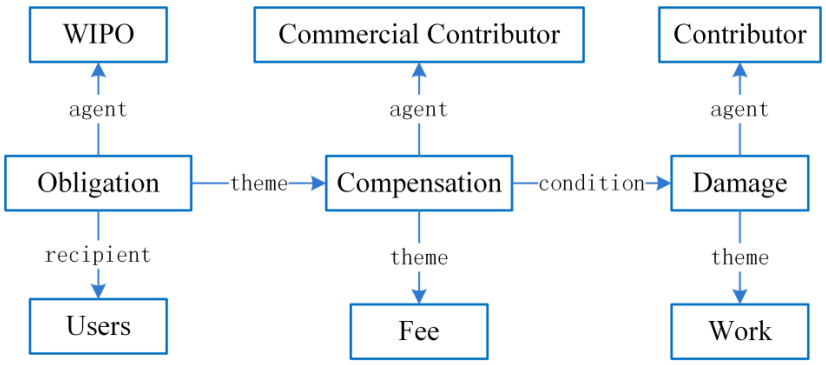

Figure 7. Instantiation of an Obligation.

4.3. Instantiation of Prohibition Pattern

As shown in figure 8, another legal regulation of Eclipse Public License v1.0, Contributors may not remove or alter any copyright notices contained within the program, means that you are not allowed to remove or alter any copyright notices if you are not a contributor.

Software license agreement domain ontology is based on the framework of three models above. Legal regulations in the software license agreement are represented by related actions and object properties, and merged into the ontology we built. However, the purpose of instantiation is to combine actions in regulations with deontic performance, which means permission, prohibition, and obligation, and present by the way of ontology that computers and humans can both understand.

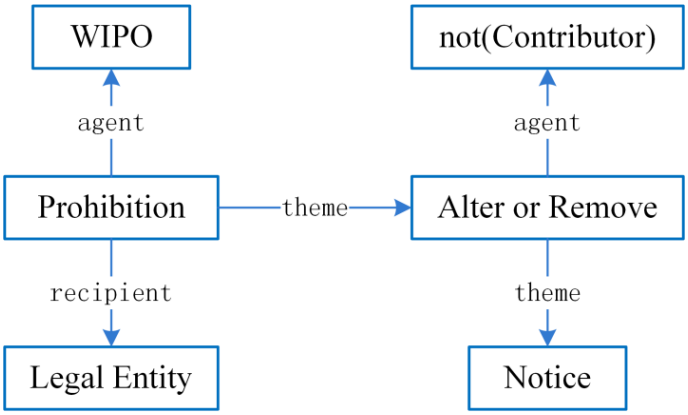

Figure 8. Instantiation of a Prohibition.

5. Evaluation of License Ontology

The previous two parts of this article series explained the modeling and instantiation of software license agreement domain ontology. On this basis, we improve the existing text summarizing technology combined with License Ontology to propose a more accurate summarizing method applying to a software license agreement.

The concepts appearing in License Ontology were weighted while calculating the frequency of each word. Because these words are more important in the domain of software license agreement, and we want to extract the summary that lets end users know which clause is permitted, obliged or prohibited. During reduplicative experiments, we adjust weight and modified our ontology continuously according to the experimental result. Consequently, the advanced text summarization tool is performing better than the original one.

To prove the accuracy and applicability of the ontology we built, classifier4J has been improved associated with software license agreement domain ontology.

As shown in figure 9, in the original algorithm, we put the first sentence where the most frequent word appears into summary. In a sense, that is right, however, after a series of experiments, we take a sentence as a unit and add the frequencies of 100 most frequent words to the sentences they belong to, and then we can get “the most frequent sentences”.

```
while (it.hasNext()) {
    String word = (String) it.next();
    sflag=isSemanticWord(word);
    int workingSentencesweight = 0;
    for (int i = 0; i < workingSentences.length; i++) {
        if (workingSentences[i].indexOf(word) >= 0) {
                workingSentencesweight = workingSentencesweight+sflag;
                resultSentences.put(actualSentences[i],
                        String.valueOf(workingSentencesweight));
        }
    }
}
//order resultSentences
resultSentences=wsortMapByValue(resultSentences);
```

Figure 9. The core code of getting “the most frequent sentences”.

Open-source summarizers apply to all domains, that is to say, it doesn’t apply to a given domain perfectly. As for a software license agreement, we cannot get an accurate summary by traditional summarizers, for the reason that license agreement differs from other text. Therefore, in the field of software license agreement, it is important to know what we should do and what not.

The improved algorithm assigns weight to the concepts in License Ontology. As shown in figure 10, when the word appears in the text once, we add N times to its frequency. The weight is formulated

based on License Ontology and it is improved continuously with the experimental result during reduplicative experiment. After that, License Ontology becomes the optimum ontology model of the domain of software license agreement. Consequently, we can find as showed below that the summary is better.

```
for (int i = 0; i < uniqueWords.length; i++) {
    if (stopWordsProvider == null) {
    // no stop word provider, so add all words
        result.put(uniqueWords[i], new
         Integer(Utilities.countWords(uniqueWords[i], words)));
    } else if (isWord(uniqueWords[i])
         && !stopWordsProvider.isStopWord(uniqueWords[i])
         &&(isSemanticWord(uniqueWords[i])==0)) {
        // add only words that are not stop words and not semanticWord
        result.put(uniqueWords[i], new
         Integer(Utilities.countWords(uniqueWords[i], words)));
    }else if (isWord(uniqueWords[i])
         && ((flag=isSemanticWord(uniqueWords[i]))!=0)){
        //semanticWord
       result.put(uniqueWords[i], new
         Integer(Utilities.countWords(uniqueWords[i], words))*flag);
    }
}
```

Figure 10. The core code of associating.

Sample dataset was taken from official website of common software in computer science that contained the software license agreements. The quality was measured by calculating the cosine similarity of the standard summary, which refers to several primary online summarizers based on Open Text Summarizer, MEAD, etc., with the summary produced by our program and subjective assessment, so we have quantification result to compare to the standard one. More formally, the cosine similarity between two summaries is given as follows:

$$cosine\ similarity(A,B) = \frac{AB}{|A||B|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n}(A_i)^2} \times \sqrt{\sum_{i=1}^{n}(B_i)^2}} \quad (1)$$

Where A is sentence weight vector of the standard summary and B is of the summary produced by Classifier4J or advanced Classifier4J. We assign different weight to different sentences according to their importance. Our evaluation of license ontology is based on this method comparing differences of two sentences.

Assumed that there are 100 sentences in document and we want 10 of them to generate a summary. Then A and B are 100-dimensional vectors. If the n-th sentence appear in the summary, the n-th vector is not 0. We assign weighting factor 3 to the two principle vectors, which means the two most important sentences. Then we assign weighting factors 2 and 1 to the other 8 vectors according to their importance of the sentence. After that, the resulting cosine similarity ranges from 0 meaning totally different, to 1 meaning exactly the same and it is more precision avoiding that result is how many sentences are the same.

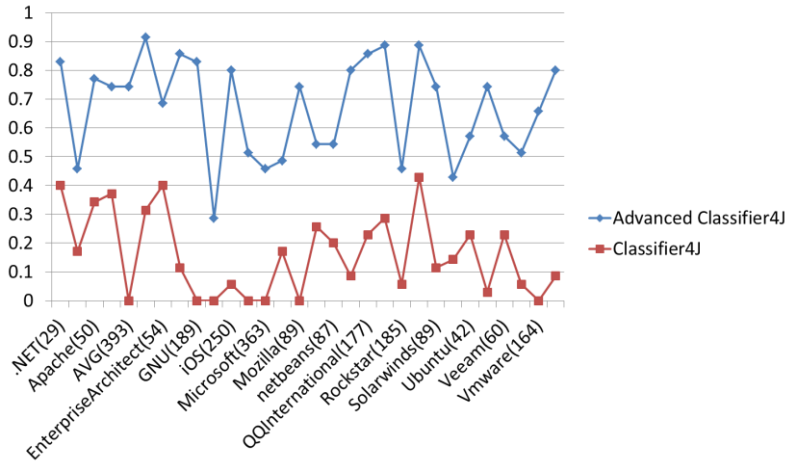

Figure 11. Comparison of Classifier4J and Method Combining with Domain Ontology.

30 documents which are presented by abscissa were tested and only 15 names of them are on the figure 11. The cosine similarity, which is presented by the vertical axis, between the summaries produced by Classifier4J and our program is given in Figure 11. There are altogether 3986 sentences in these documents. From the experimental data we can find that it enhances a lot after combining with domain ontology. Before the improvement of the algorithm, the summaries of Classifier4J are very different from the standard summaries. However, after combining with License Ontology, the average cosine similarity increases by 51 and is close to 70 percent. We can find the accuracy and applicability of the ontology we build. Unfortunately, the length of the license agreement really affects the accuracy.

However, it's worth mentioning that the main difficulty in evaluation comes from the impossibility of building a fair gold-standard. Furthermore, it is also very hard to determine what a correct summary is, because different users would choose different sentences as important information. Even a particular person may choose different sentences at different times. Hence, the quantitative evaluation might not be the only way to evaluate summaries. We also refer to subject assessment from researcher during the evaluating and analyzing of experimental results.

6. Conclusions and Future Work

The Software License Agreement Domain Ontology is an ontology based on the software license agreement framework that is capable of modeling complex value chains and legal regulations. During the construction of the domain ontology, this paper gives three top models associate with software license agreements to enrich domain ontology. It is pleased to see that the computer can understand what legal regulations say exactly.

Our approach to construct ontology is modular in that distinct sub-components of the ontology can be independently developed as we outlined in section 3. First of all, it provides a complete role model that takes into account the different roles of person and action. Second, there is the deontic model that captures concepts from the underlying legal regulations and gives shape to the third component, the action model. The latter includes the actions governed by the software license agreement that can be performed by the concepts defined in the role model and deontic model, which are associated with a series of object properties so that the ontology can illustrate what regulations express. However, for certain licenses, event patterns are essential to state what is permitted, prohibited or obliged by a license. The proposed actions and roles are used to model event patterns in the software license agreement. The purpose of instantiation is to combine actions in regulations with deontic performance and presented by the way of ontology.

Moreover, combined with text summarization, a more accurate summarizing method is creating applying to software license agreement. To prove the accuracy and applicability of the ontology we build, classifier4J has been improved associated with software license agreement domain ontology.

Protégé is used as our ontology editor and we build our domain ontology using Web Ontology Language. We use data from "Computer Software Protection Ordinance of the People's Republic of China", "Copyright Law of the People's Republic of China", "Patent Law of the People's Republic of China", and software license agreements like Eclipse, Apache, Microsoft, etc.

In a sense, the construction of domain ontology is endless. This paper aims to propose an informative, descriptive and scalability software license agreement ontology. The future work concentrates now on extending the domain ontology and offering a recommendation system that is easy to operate as shown before. This recommendation system will help end users understand licenses with less trouble, which is a sincere way to serve the users.

7. References

[1] Web Ontology Language (OWL) W3C recommendation, www.w3.org/2001/sw/wiki/OWL, last accessed 2017/3/11

[2] Chandrasekaran B, Josephson J R and Benjamins V R 1999 what are ontologies, and why do we need them? IEEE Educational Activities Department

[3] Lu W and Ikeda M 2005 An Intention-oriented model of copyright law for e-learning: international semantic mapping of copyright laws based on a copyright ontology. In: Conference on Towards Sustainable and Scalable Educational Innovations Informed by the Learning Sciences: Sharing Good Practices of Research, pp.757-760

[4] Dalal V and Malik L G 2013 A survey of extractive and abstractive text summarization techniques. In: Sixth International Conference on Emerging Trends in Engineering and Technology (ICETET). IEEE, pp:109-110

[5] Automatic summarization 2017 https://en.wikipedia.org/wiki/Automatic_summarization

[6] Jones, K. S 2007 automatic summarising: the state of the art. *Information Processing & Management An International Journal* **43(6)**, 1449-1481

[7] Mccarty L T 1989 a language for legal discourse I. basic features. In: International Conference on Artificial Intelligence and Law. pp.180-189

[8] Ontology, https://en.wikipedia.org/wiki/Ontology

[9] Valente A, Breuker J and Brouwer B 1999 Legal modeling and automated reasoning with ON-LINE. *International Journal of Human-Computer Studies* **51(6)**, 1079-1126

[10] Breuker J, Valente A and Winkels R 2004 Legal ontologies in knowledge engineering and information management. *Artificial Intelligence and Law* **12(4)**, 241-277

[11] Hoekstra R, Breuker J, Bello M D, et al 2007 The LKIF core ontology of basic legal concepts. In: The Workshop on Legal Ontologies & Artificial Intelligence Techniques June. DBLP. pp.43-63

[12] Wyner A and Hoekstra R 2012 A legal case OWL ontology with an instantiation of Popov v. Hayashi. *Artificial Intelligence and Law* **20(1)**,83-107