Exploiting Syntactic and Semantic Information for Relation Extraction from Wikipedia

Dat P.T Nguyen[1], Yutaka Matsuo[2], and Mitsuru Ishizuka[1]

[1] The University of Tokyo, 7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8656, Japan
nptdat@mi.ci.i.u-tokyo.ac.jp,
WWW home page: http://www.miv.t.u-tokyo.ac.jp/HomePageEng.html
[2] National Institute of Advanced Industrial Science and Technology
Sotokanda 1-18-13, Tokyo 101-0021, Japan

Abstract. The exponential growth of Wikipedia recently attracts the attention of a large number of researchers and practitioners. One of the current challenge on Wikipedia is to make the encyclopedia processable for machines. In this paper, we deal with the problem of extracting relations between entities from Wikipedia's English articles, which can straightforwardly be transformed into Semantic Web meta data. We propose a method to exploit syntactic and semantic information for relation extraction. In addition, our method can utilize the nature of Wikipedia to automatically obtain training data. The preliminary results of our experiments strongly support our hyperthesis that using information in higher level of description is better for relation extraction on Wikipedia and show that our method is promising for text understanding.

1 Introduction

Wikipedia [3] has been emerging as the world's largest encyclopedia. The term wiki indicates that the online encyclopedia can be freely contributed by anyone who can access to its site. Although it started from 2001, English version of Wikipedia contains 1.3 million articles and averagely 2,106 articles are newly created a day as of June 2006 [4], which implies that it is growing as an exponential pattern [5]. Since the encyclopedia is managed by Wikipedia Foundation, an international non-profit organization, and a great number of collaborators in the world under some international projects, its articles are continuously edited and developed. Therefore, its content is quite reliable regardless its openness.

Although Wikipedia contains an invaluable source of information, the usage of Wikipedia is currently limited to only for human readers. The most important work that computer systems can support in using Wikipedia source is searching and showing interested articles to readers, without offering information automatically gathered from various related articles [1]. The limitation is due to the low formality of Wikipedia's content for machine to be able to process. In particular, the articles in Wikipedia are written in natural languages and thus they prevent machines from processing their content semantically. To improve the usage of Wikipedia, it is necessary to represent Wikipedia's knowledge in the more formal format which supports machine-processable.

One can imagine a system which is able to receive machine-processable knowledge from Wikipedia as the data source, and offers a greater satisfaction of information need to the users. This goal is within the mission of Semantic Web [2], a well-known infrastructure for the next generation of World Wide Web. The Semantic Web is based on RDF [3], a representation language using Notation 3 or N3 [4]. We follow the formalism of Semantic Web in which we structure Wikipedia's content as a collection of statements. Each statement consists a subject, a predicate and an object. For example, the statement (Microsoft, *Founder*, Bill Gates) will represent the knowledge of the sentence: "Bill Gates is one of the founders of the Microsoft Corporation". The statements can then be straightforwardly transformed into RDF format that in turn serves as machine-processable knowledge base. The quasi-exponential growth of Wikipedia allows the structured Wikipedia to become the world's largest directory

[3] http://www.wikipedia.org/
[4] http://stats.wikimedia.org/EN/TablesWikipediaEN.htm
[5] http://en.wikipedia.org/wiki/Wikipedia:Modelling_Wikipedia's_growth

of important people, companies; the largest collection of famous products and many other essential machine-processable resources.

In this paper, we describe a novel method to deal with relation extraction problem for English version of Wikipedia encyclopedia. Particularly, our method attempts to integrate syntactic and semantic information of text to form an unified structure. The method then decomposes the structure into subsequences and mine the frequent ones with the aim to capture the key patterns for each relationship. We also make use of the Wikipedia's natures to automatically obtain training data, which gives our system high portability for new relationships with no human labor is needed.

2 Problem Statement

In this section, we define our problem along with some assumptions.

We aim at extracting binary relations between entities from English version of Wikipedia articles. An entity pair is defined as a 2-tuple of (e_p, e_s) where e_p and e_s are entities which may be PERSON, ORGANIZATION, LOCATION, TIME or ARTIFACT. We further define binary relation as a triple (e_p, *rel*, e_s) in which, e_p and e_s are entities and *rel* indicates the directed relationship between e_p and e_s. For example, (Microsoft, *Founder*, Bill Gates) is a relation indicating that Bill Gates holds a Founder relationship on Microsoft. A full list of interested relations is described in Table 1A.

Our system is given Wikipedia text and should return a set of triples as extracted relations. We follow [5] to define the entity mainly discussed in an article as *principal entity*, and other mentioned entities in the same articles as *secondary entities*. We assume that interested entities in this problem should have a descriptive article in Wikipedia to avoid the errors of entity recognition. Thus, all the secondary entities should be attached a unique hyperlink to a Wikipedia's article. Generally, the identifier of an entity is defined as the URL address to its appropriate article.

Because of the nature of Wikipedia, most of the sentences in an article discuss its principal entity. Furthermore, it is likely that the relation between a pair of secondary entities, if any, may also be discussed in their own articles. For these reasons, our system predicts only the relations between the principal entity and each mentioned secondary entity in an article. As one more assumption, the relationship between an entity pair can be completely expressed in one sentence. So that, for an article, only the sentences that contain a principal entity and a secondary entity are necessarily to be analyzed, all the others can be eliminated. This leads to the necessity of Sentence Detector module which appears in Section 4.1.

3 Related Works

Some important works on relation extraction by learning surface text were introduced in [6], [7] and [8]. In these researches, the authors conducted experiments on web data which is so abundant that it enables their systems to obtain easy patterns. The systems then learn such patterns mostly based on lexical information. Although these systems apply boot strapping strategies to explore more data from the patterns learned in previous steps, they hardly move to a new domain. Furthermore, Ravichandran and Hovy claimed in [8] that their method cannot handle the long-distance dependencies. Thus, the above methods may fail in extracting relations from the Wikipedia source which is observed to be more formal, complex but not abundant.

Recently, the authors in [9] present a kernel method to classify relationships of entity pairs. They propose a matching method for dependency paths to estimate their similarity. Their method makes the assumption that paths with different lengths tend to express different relationships. The kernel function will assign 0 value to the pair of paths in this case. Additionally, in case the paths satisfy the condition of length, the kernel function estimates the similarity of the paths by multiplying the matching result of corresponding positions, which requires the paths to be well aligned. Their method may be more efficient if the assumption can be relaxed. Our work attempt to overcome the problem by matching from the decomposed subpaths, which enable the path to match to other paths regardless the difference in length.

Culotta et al. [5] present a probabilistic model to integrate extraction and mining tasks performed on biographical text of Wikipedia. To avoid the suffering from the errors accumulated through the traditional pipeline, they formulate the relation extraction problem into

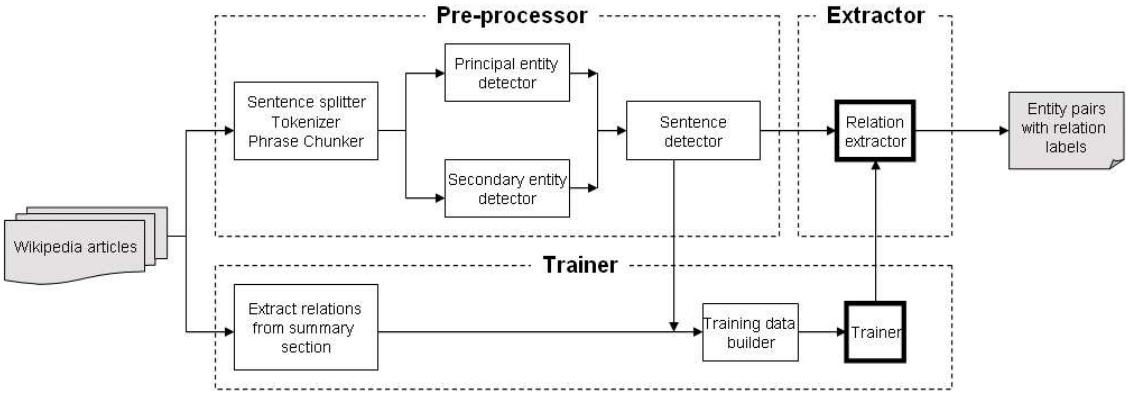

Fig. 1. System framework

sequence labeling problem which then is solved by Conditional Random Field [10]. Their supervised method uses both contextual and relational information to enable the two tasks support each other to improve the performance of the whole integration system. They solve the problem of label the sequence

4 Extract Relations from Wikipedia

In this section, we describe our methods to extract relations between entities from Wikipedia text. Section 4.1 will explain the framework of our systems along with some pre-processing steps. The core methods are then described in Section 4.2 and 4.3, in which one uses only syntactic information and the other utilizes the integration of syntactic and semantic sources. We attempt to claim that using more information from higher level of description can reduce the variation of surface text. Consequently, the structure of the text we gain from such information will semantically reflect the behaviors of the concepts in sentences.

4.1 Relation Extraction Framework

Figure 1 illustrates our framework for relation extraction. First of all, articles should be processed to remove the HTML tags, extract hyperlinks which indicate to other Wikipedia's articles. To start the pre-processor, they are submitted to a pipeline including a sentence splitter, a tokenizer and a phrase chunker. We use OpenNLP [6] tool set to analyze the text. The articles are then parallelly processed to anchor all the occurrences of the principal entity and the secondary entities. The secondary entity detector simply labels the appropriate surface text of the hyperlinks returned in the previous step as secondary entities. The algorithm to detect principal entity will be describe in the next part of this section. After the entities are anchored in the text, sentences that include at least one pair of the principal entity and a secondary entity. Both trainer and extractor share the same pre-processor. The trainer receives articles with HTML tags to identify the summary section and extract ground true relations (correct relations) from the section. Sentences, received from the pre-processor, that contain entity pairs from ground true relations are selected and submitted to training data along with the descriptive labels of the between the pairs. The trainer will learn the key patterns with respect to each relation from training data and output the learned data to the extractor. For each sentence and the entity pair on it, the relation extractor will identify the descriptive label based on the learned data. It then outputs the final results in form of relations with descriptive labels.

In the remainings of this section, we will supply more details for the principal entity detector, the sentence detector, the module to extract ground true relations from summary section and the training data builder. The two next sections will describe our core methods for the trainer and the relation extractor.

[6] http://opennlp.sourceforge.net/

Article	Referents
Bill Gates	[NP Bill/NNP Gates/NNP] [NP Gates/NNP] [NP The/DT Gates/NNP] [NP William/NNP H./NNP Gates/NNP] [NP he/PRP] [NP him/PRP]
Microsoft	[NP It/PRP] [NP Microsoft/NNP] [NP The/DT Microsoft/NNP Corporation/NNP] [NP that/DT Microsoft/NNP] [NP the/DT company/NN]
Microsoft Windows	[NP Microsoft/NNP Windows/NNP] [NP Microsoft/NNP] [NP Windows/NNP] [NP it/PRP] [NP the/DT Windows/NNP]

(A)

(B)

Fig. 2. Referents of some principal entities (A) and the summary section in Wikipedia's Microsoft article (B)

Principal Entity Detector The outputs of the chunker from each article are then processed by a module to detect all the appearances of the principal entity in the article. Analysis on Wikipedia text helps us explore the following the characteristics of Wikipedia articles:
-An article mainly discusses its principal entity so that most of the pronouns refer to the principal. For example, the pronoun "it" and "he" mainly refer to "Microsoft" and "Bill Gates" in the corresponding articles respectively.
-The first sentence of the article is often used to define the principal entity. For example, we respectively found the sentences "*The Microsoft Corporation*, commonly known as just Microsoft, (NASDAQ: MSFT, HKSE: 4338) is a multinational computer technology corporation..." as the first sentences of the "Microsoft" articles.

Based on the above characteristics, we propose a technique to identify a set of referents to the principal entity, denoted by D. We classify the referents in D into three types: (1) pronoun ("he", "him", "they", "them"...) (2) proper noun (e.g., Bill Gates, William Henry Gates, Microsoft,...) (3) common nouns (the company, the software,...). Steps to collect the referents for D include:

(i) Start with $D = \{\}$

(ii) Select the first two chunks: the proper chunk of the article title and the first proper chunk in the first sentence of the article if any. We define a proper chunk as a chunk that contains at least a proper noun, word with NNP or NNPS tag. We call the first proper chunks selected so far (up to two) in D as names of the principal entity.

(iii) If D contains no items, return D and stop the algorithm. Otherwise, continue.

(iv) Get a list of all other proper chunks of the articles. For each proper chunk p, if p is derived from any of the names, then $D \leftarrow p$. We call a proper chunk p_1 is derived from a proper chunk p_2 if the set of all proper nouns of p_1 is a sub set of all proper nouns of p_2.

(v) From the article, select c as the most frequent pronouns, find c' as its equivalent pronoun and add them to D. For example, 'he' is equivalent to 'him', 'she' is equivalent to 'her' ...We call c and c' pronoun referents.

(vi) Select all the chunks with the pattern $[DT\ N_1\ ...\ N_k]$ where DT is a determiner and N_k is a common noun. Only those chunks which appear more frequently than the pronoun referents are added into D.

Figure 2A shows some sample referents extracted for several articles by the above technique. Supported by the nature of Wikipedia, our technique performs better than those of the coreference tool in LingPipe library [7] and coreference package in OpenNLP tool set. All the occurrences of the collected referents are labelled as principal entity.

Sentence Detector Receiving output from the two previous detectors, this module selects sentences that contain at least one occurrence of the principal entity and one occurrence of

[7] http://www.alias-i.com/lingpipe/index.html

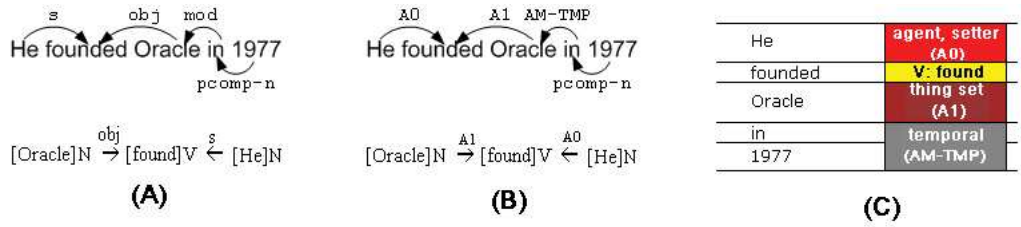

Fig. 3. Syntactic (A), semantic (C) and integrated representation (B) of a sample sentence

a secondary entity. Each pair of occurrences, one for the principal and one for a secondary entity, becomes a relation candidate. So, there may be more than one relation candidate on a sentence.

Extract Relation from Summary Section Although Wikipedia's articles mainly provide information in form of natural language text, those articles about famous and important entities contain summary information. For example, one can find the summary section in Microsoft article as shown in Figure 2B. In the view of relation extractor, we can automatically obtain some interested relations from the box, such as (Microsoft, *Foundation*, Albuquerque), (Microsoft, *Founder*, Bill Gates), (Microsoft, *CEO*, Steve Ballmer)...

In this research, we exploit the seed relations extracted from the summary sections to create training data. Therefore, our method requires no human efforts in building training data as in traditional supervised learning methods. Consequently, it is highly portable for new relations in different domains without any human labor provided that such relations appear in the summary sections of some articles in the whole Wikipedia archive. From here, ground true relation refers to the surely correct relations obtained by this way.

Training Data Builder We define a training sentence as a six-tuple: (s, l_1, r_1, l_2, r_2, rel) where s is the sentence itself, (l_1, r_1) and (l_2, r_2) indicate the token-based boundaries of the principal entity and secondary entity, and *rel* indicates the relation label between the entity pair. With a sentence s selected from the above procedure, we receive the identifiers of the entities and search for a ground true relation r such that $r.e_p$ is the identifier of the principal entity and $r.e_s$ is the identifier of the secondary entity. Then, we create a new training sentence from s and r.

For a relationship r, the purpose of building training data is to collect the sentences that exactly express r. To reduce noise in training data, it is necessary to eliminate the pairs from the ground true set which hold more than one relation.

4.2 Learning Patterns with Dependency Path

In this section, we will explain our first method to extracting relation using syntactic information. The training sentences received from the training data builder contain at least one entity pair with descriptive label. These sentences will be analyzed by Minipar dependency parser [11] to derive a dependency tree. Borrow the idea in [9], we assume that the shortest dependency path tracing from the principal entity through the dependency tree to the secondary entity gives the syntactic structure expressing the relationship between the entity pair. Figure 3A and 4A show examples of dependency trees and dependency paths between the entities.

One of the challenge for this problem is due to the wide variation of the surface text. The challenge comes from the combinatorially explosive number of ways to express one idea. For example, although the sentence "Adobe Systems is an American computer software company that was founded in December 1982 by John Warnock and Charles Geschke" and the sentence in Figure 4 express Founder relationship, the surface text is totally different. However, the sentences seem to be converged when we move from lexical level to syntactic level. In other words, the syntactic analysis of the sentences enable us to reduce the variation of the sentences. A closer analysis of the dependency path between "Adobe Systems" and "John Warnock"

VI

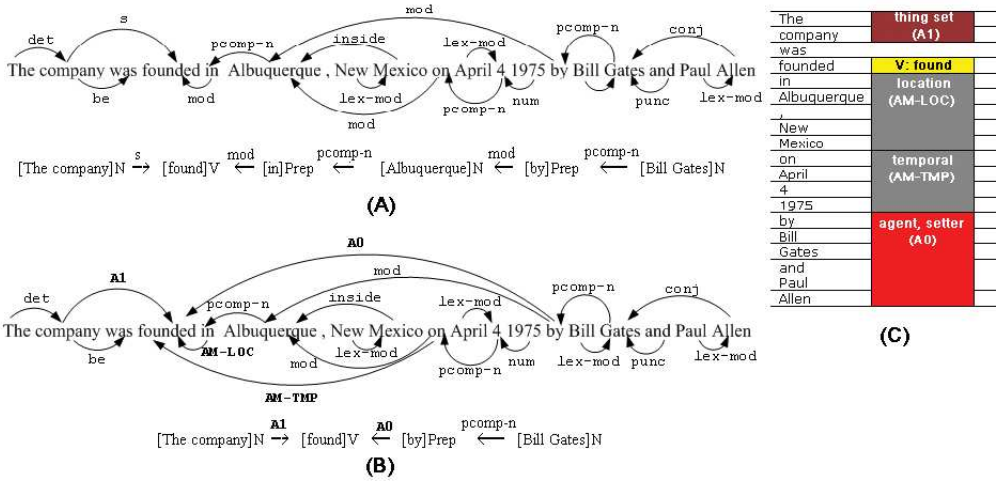

Fig. 4. Syntactic (A), semantic (C) and integrated representation (B) of another sentence

and the dependency path in Figure 4A, we can observe that they share a common segment of path "[found]V $\leftarrow^{mod}$ [by]Prep $\leftarrow^{pcomp-n}$ N".

Our idea is to learn such key patterns from the dependency paths with respect to a relation. In particular, we transform the dependency paths into sequences which are in turn decomposed into subsequences by the below mining technique. A collection of training sentences with respect to a relation has a corresponding collection of subsequences with respect to the same relation. From the subsequence collections of various relations, we can identify the frequent subsequences for each relation. During testing, dependency path between an entity pair in a novel sentence is also converted into sequence and match with the previously mined subsequences. Sequence A matches sequence B if and only if B is a subsequence of A. The more frequent subsequences of a relation it matches, the more likely that the original sentence express the relation between the. From now, we call pattern and subsequence interchangeably.

Sequential Representation of Dependency Path We transform a dependency path into a sequence as follows:
(i) For each word excluding the first and the last ones in the dependency path, we consider the combination of its base form and its Part-Of-Speech (POS) as an element of the sequence.
(ii) For the first and the last word, only the POS is concerned as a sequence element.
(iii) For a dependency relation, a pair of direction and relation label is considered as an element of the sequence.
Figure 5 gives an example of the sequential representation.

Learning Key Patterns as Mining Frequent Sequence PrefixSpan, which is introduced in [12], is known as an efficient method to mining sequential patterns. Given a sequence database in which each sequence contains a set of itemsets, PrefixSpan will find all the subsequences appearing more frequently than a support threshold. A sequence s=$< s_1 s_2 ... s_n >$ is called subsequence of a sequence p=$< p_1 p_2 ... p_m >$ if there exists a set of integers $1 <= j_1 < j_2 < ... < j_n <= m$ such that $s_1 \subset p_{j_1}$, ..., $s_n \subset p_{j_n}$. As mentioned above, our problem of learning key patterns is casted to a special case of the above problem in which all the itemsets of a sequence contain only one item. In this research, we use the implementation tool [8] of PrefixSpan developed by Taku Kudo. The tool also returns the support of each frequent subsequence, which is defined below. From here, sequence database denotes the set of sequences converted from dependency paths with respect to a relation.

Weighting The Patterns It is necessary for each mined pattern to be assigned a weight with respect to a relation for estimating the relevance of the pattern to the relation. The weight should incorporate the following factors:

[8] http://www.chasen.org/ taku/software/prefixspan/

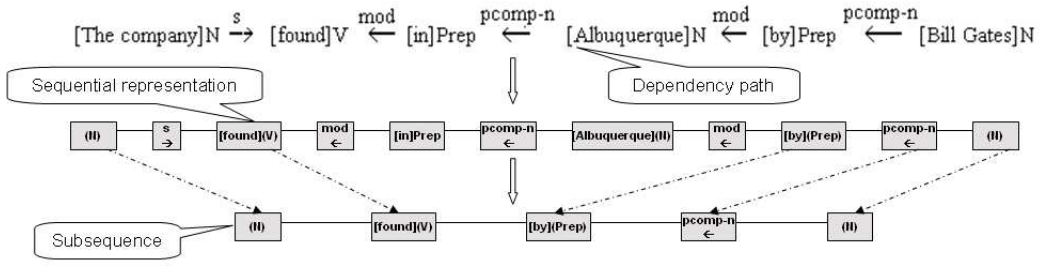

Fig. 5. Sequential representation of a dependency path

(i) *Length of the pattern*: if two paths share a long common sub-pattern, it is more likely that the paths express the same relationship.

(ii) *Support of the pattern*: is the number of sequences that contain the pattern. Indeed, the support is also the number of occurrences of the pattern in the sequence database. Thus, it is more likely for the pattern with high support to be a key pattern of the relation.

(iii) *Amount of lexical information*: although the sequences contain both words and dependency relations from the original dependency path, we found that the items coming from words is more important than those coming from dependency relations. Let us consider the following sentences: "He is the founder of the company A" and "He is the director of the company A". Although all the dependency relations in the sentences are the same, they express totally different relationships due to the words "founder" and "director".

(iv) *Number of sequence databases in which the pattern appear*: this factor is similar to the concept of Inverted Document Frequency in TF-IDF. If the pattern can be found in various sequence databases, it is more likely that the pattern is common. In other words, it may not be a key pattern with respect to any relation even it is frequent.

From the above analysis, we calculate the weight of a pattern with respect to a relation r as the following formula:

$$w_r(p) = \frac{irf(p) \times support_{D_r}(p) \times l(p) \times e^{lex(p)}}{|D_r|}$$

where

- D_r is the sequence database of the relation r.
- irf(p) is the Inverted Relation Frequency of p that is calculate by $log(\frac{|R|}{|M(p)|})$ where R is the set of target relation and M(p) is set of sequence database in which p occurs.
- $support_{D_r}(p)$ is the support of p in D_r
- l(p) is length of the pattern p
- lex(p) is the number of word-based items in p

Relation Selection Given a novel sentence and the anchors of an entity pair in it, we will predict the appropriate relationship of the entities. We first extract the dependency path P between the entities, transform P into sequential pattern and then accumulate the scores of its subsequences with respect to separate relation r as the following:

$$L_r(P) = \sum_{p \in S(P)} w_r(p)$$

where

- $L_r(P)$ likelihood score to say that P express relation r
- S(P) set of all subsequences of the sequential representation of P

The appropriate relation should be the one giving highest score to P.

$$R = \operatorname*{argmax}_r L_r(P)$$

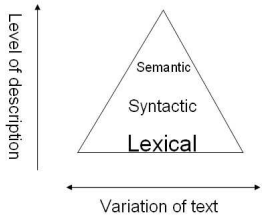

Fig. 6. The convergence of the variation of semantically analogous sentences when moving to higher level of description.

4.3 Learning Patterns with Dependency Path and Semantic Role

We continue to form the thesis that the variation of semantically analogous sentences will be converged when we move to higher levels of description as illustrated in Figure 6. Let us consider some examples shown in Figure 3 and 4. The two sentences express the Founder relationship between a company and a person but in different surface text. The syntactic representation of the sentence in Figure 3A captures the person as the *subject* (abbreviated as 's') and the company as the *object*. Instead, syntactic representation in Figure 4A shows that the company is *subject*, since the sentence is in passive voice. Thus, no common subpattern except the one contain only three nodes "N", "[found]V" and "N" when compare the two dependency paths. As the results, it is difficult to recognize that the two sentences give the same relationship. Stop by the syntactic level, the representation just captures the subject and object as syntactic roles of the sentence. It is impossible to semantically know which instance perform the action. So, when the syntactic structure of the sentence changes, the syntactic representation is modified.

It is more reasonable if we can capture the stable roles of the companies and the persons of the sentences in the context of 'founding' regardless of the variation of the surface text, given by the change of the voice in this case.

The above thesis suggests us to use *frame semantics* theory, a semantic representation which is promising for Natural Language Understanding [13], to represent the text. A frame is considered as a schematic representation of situations in which relationships between participants, props and other conceptual roles are shown. In PropBank project [14], the structure of the frame and its participants are called Predicate-Argument (PA) structure. Figure 3C illustrates the PA structure of the a sentence, in which "found" is the predicate; "He", "Oracle" and "in 1977" respectively play "agent", "thing set" and "temporal" roles in the context.

The development of large corpora such as PropBank [14] and FrameNet [15] enables the process of filling PA structures with constituents from text, which is well-known as Semantic Role Labeling (SRL) task [16]. In this research, we use the SNoW-based Semantic Role Labeler [17], a state-of-the-art in SRL task. The tool conforms the definition of PropBank and CoNLL-2004 shared task [9] on SRL .

Since the SRL task just labels roles to constituents or phrases without indicating which primitive concept playing the role, we still use dependency parsing information to further analyze the phrases. We combine the two information sources by integrating semantic role information into dependency parse tree of a sentence. The steps for the integration are as follows:

(i) For each relation between a predicate P and its role R in the PA structures of the sentence, identify the headwords of the two phrases.

(ii) If there exists a dependency relation between the headwords, replace it by a new relation. Otherwise, create the new relation between the headwords. The new relation is directed, receiving the headword of P as its head, headword of R as its tail and R as its label.

Some examples in Figure 3 and 4 illustrate the integration process, that is the dependency trees in (A) are augmented by PA structures in (C) to obtain an integration trees in (B). In this method, only the dependency trees are augmented by PA structures. All the other steps are the same to those of the previous method in section 4.2.

[9] http://www.lsi.upc.edu/ srlconll/

CEO	Spouse
Founder	Birth date
Chairman	Birth place
COO	Foundation
President	Product
Director	Location
Vice Chairman	

(A)

	RetRel	Ret	Rel	Prec	Rec	F1
B0	1,962	5,975	5,975	0.3284	0.3284	0.3284
B1	2,665	5,975	5,975	0.4460	0.4460	0.4460
Dep	2,970	5,257	5,975	0.5650	0.4971	0.5288
DepSRL	3,449	4,991	5,975	**0.6910**	**0.5772**	**0.6290**

(B)

Table 1. List of target relations (A) and the result table (B) in which the columns *RetRel*, *Ret* and *Rel* can be considered as the number of correctly retrieved relations, the total number of retrieved relations and the number of relevant relations respectively as in IR field.

5 Experimental Settings

5.1 Data

In this research, we evaluate our methods on the list of relations listed in Table 1A

We perform our experiments on real Wikipedia data dumped [10] on Aug 10, 2006 . To enable the automatic evaluation, we interest only the articles including the summary sections that provide at least one target relation in the above list. Wikipedia defines templates for the summary sections. For example, most of the general company articles will contain the template called Infobox_Company, person articles may contain Infobox_Senetor, Infobox_Celebrity... We first compile a list of appropriate template names of the target relations. Then, we select all the articles that include our selected templates. In this experiment, 6,125 articles (corresponds to 6,125 entities) are selected, along with 21,356 ground true relations and 112,864 relation candidates distributed in 48,138 sentences.

5.2 Baseline Systems

To prove the claim that using syntactic and semantic information may improve the performance of relation extraction, we develop two baseline systems. Both of the systems use Bag Of Words model, in which only words themself are concerned. In the first system, we assume that words in the between express the relationship of the entity pair. The system also performs all the steps as described in Section 4.1. Then the trainer will extract all the words in the between of the principal and secondary entities. So, each relation has a list of relevant words. TFIDF score is then calculated for each word with respect to a relation. Actually, the training process is aiming at identifying keywords for each relation. When the system faces a new entity pair in a new sentence, it also extracts the words in the between and accumulates the TFIDF scores of the words for each relation. Finally, it chooses the relation that gives the highest score and assigns to the entity pair.

The only difference between the two baseline systems is that the second one use dependency parse tree to eliminate the irrelevant words. In other words, only words on the path from the principal entity to the secondary entity are extracted instead of all the words in the between.

5.3 Evaluation

Usually, full evaluation of a relation extraction system requires a human annotated relation set which in turn may require huge amount of human labor. In this research, we utilize the ground true relations as mentioned in Section 4.1 to evaluate our method. Although the data is automatically derived from Wikipedia, it is highly correct because it is created by human editors and contributors of Wikipedia. The only flaw of using this dataset for evaluation is due to its coverage. In particular, it does not contain all the relations existing in the natural language text of the articles on which our systems work and this dataset may contain relations absent from the article text. However, we compare all the methods in the same condition with the same manner that described below. So, the results in the following section are still strongly believable and the comparison is credibly fair.

[10] http://download.wikimedia.org/enwiki/20060810/

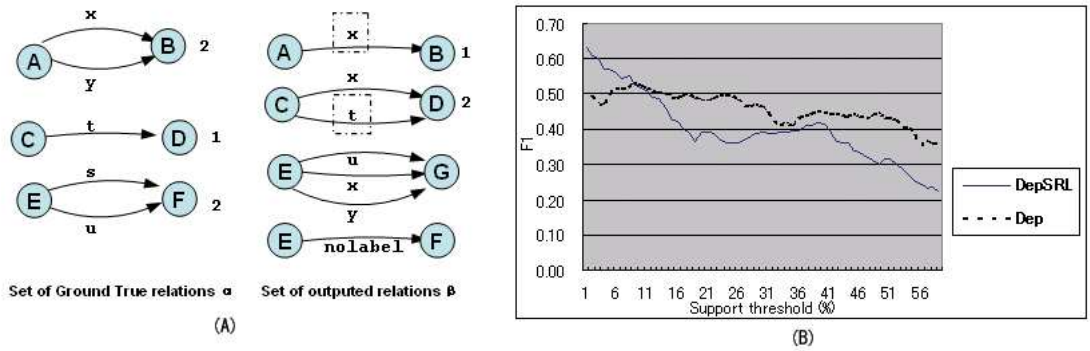

Fig. 7. (A) An example for evaluation setting and (B) curves to compare the system using only syntactic information (Dep) and the system using syntactic and semantic information (DepSRL)

We follow precision and recall metrics from Information Retrieval to evaluate in this task. Firstly, we attempt to estimate the list of retrieved relations, list of relevant relations and list of retrieved relations that are correct from the result of our systems and the set of ground true relations. Secondly, we calculate the precision and recall as usual. The followings describes our setting for evaluation:

Let α and β be the set of ground true relations and the set of relations outputed from the system respectively. Please note that β includes all the relation candidates returned by the Sentence Detector and their descriptive labels. Consider a relation $r \in \beta$, if $r.rel$='nolabel',the system returns no relationship between $r.e_p$ and $r.e_s$, otherwise a relationship between the entity pair is recognized. We assume that if an entity pair appears in both α and β, then all the ground true relations between it in α are also expressed in text and all the recognized relations between it expressed in text should be in α. So, we define the following set of relations:

•$\beta' = \{r | r \in \beta \wedge r \in \alpha\}$. As the definition implies, this is the set of correctly extracted relations. $|\beta'|$ corresponds to the numbers in *RetRel* column in Table 1B.

•$\beta'' = \{t | t \in \beta \wedge t.rel \neq nolabel \wedge (\exists l \in \alpha : t.e_p = l.e_p \wedge t.e_s = l.e_s)\}$, set of relations in β between the entity pairs which are also contained in α. $|\beta''|$ corresponds to the numbers in *Ret* column in Table 1B.

•$\alpha'' = \{t | t \in \alpha \wedge (\exists l \in \beta : t.e_p = l.e_p \wedge t.e_s = l.e_s)\}$, set of relations in α between the entity pairs which are mentioned in β. $|\alpha''|$ corresponds to the numbers in *Rel* column in Table 1B.

Then, we calculate precision and recall:

$$Precision = \frac{|\beta'|}{|\beta''|}, \qquad Recall = \frac{|\beta'|}{|\alpha''|}$$

The example illustrating our evaluation method is given in Figure 7. Here, the entity pairs *(A, B), (C, D), (E, F)* appear in both of the sets α and β. Thus all of the relations in α which are held by the pairs are counted for the relevant set, and all of the relations in β which are held by the pairs are counted for the retrieved set. In this example, only *(A, x, B)* and *(C, t, D)* are correctly retrieved although the system recognized 6 relations. However, according to our evaluation setting, relevant set contains 5 relations, retrieved set contains 3 relations. Therefore, this system obtains 2/3=0.67 and 2/5=0.4 as precision and recall scores respectively. The relations between *(E, G)* cannot be evaluated.

5.4 Results

Table 1B shows the results of four systems under the same dataset and evaluation setting described in the above section. We use 5-fold cross validation to test our systems. "B0"denotes the baseline systems using trival Bag-Of-Words model while the system denoted by "B1" uses BOW model together with dependency parsing to remove irrelevant words. "Dep" and "DepSRL" denote the system mining frequent patterns from dependency paths and the system

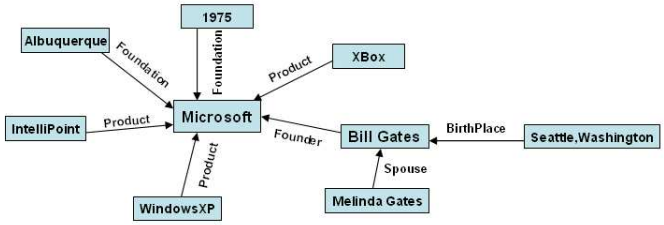

Fig. 8. Some successfully extracted relations from our best system (DepSRL with 2% as support threshold)

mining frequent patterns from dependency paths augmented with PA structures described in Section 4.2 and 4.3 respectively. Please note that the above numbers of relations reflect only the overlapping part between the relations outputed by the systems and the ground true relations. Actually, our systems extract more relations which are not listed in the result since they are far from evaluation.

As mentioned in Section 4.2, PrefixSpan algorithm accepts a support threshold as an parameter to set the minimal occurrences of the mined subsequences. Thus, "Dep" and "DepSRL" systems depends on the support threshold. The result of the "Dep" system in Table 1B obtained from the system with 10% as support threshold while the "DepSRL" system obtains the best result at 2% of support threshold. We also report the comparison of these two systems as in Figure 7 when varying the support threshold. The improvement of the systems when support thresholds decrease can be explained that small thresholds enable more subsequences to be mined, which include the key subsequences of the relationships. In other words, high thresholds may lose some important subsequence. This implies that the some featured sentences for some relationships may be rare in training data. The system using more semantic information outperforms the system using only syntactic information with reasonable support threshold. Figure 8 shows some relations extracted by our system.

From the experimental results, we conclude that the more syntactic and semantic information we use, the better result we can obtain.

6 Conclusions and future works

We have shown a method to extract relations between entities from Wikipedia text. The key innovations of our method include (1) a newly proposed structure for text incorporated from syntactic and semantic source, which can capture the behaviors of concepts in sentences regardless the distance from them to their participants (2) a technique to obtain the key sub-structures for relations. Although it is still far from the ultimate goal, our method can be considered as a step towards deep analysis of natural language text. We have also proposed the usage of Wikipedia's summary sections to make our system easily portable for novel relationships.

We believe that the current setting can be improved because some relations may be lost due to the limitation of secondary entities in hyperlinks. As the future work, starting from the chunks of text under hyperlinks, we will extend the appearances of secondary entities. This may lead to the improvement of recall.

Additionally, we intend to make use of more Wikipedia's characteristics, such as the Wikipedia's category hierarchy and link structure. By using those kinds of information, we can recognize the type of entities to make the contraints on available relationships an entity may receive, which in turn improves relation discriminating process.

Furthermore, we plan to extend the usage of various kinds of summary sections to enrich the training data. By doing this, our model can estimate the score for key patterns more correctly. Also, a way of reducing noise in training data to improve the accuracy should be investivated. The motivation is that it is difficult to choose the sentences for the training set that purely express only one relationship although the automatically derived relations from the summary sections of Wikipedia articles are correct.

Finally, we plan to combine our method with a data mining module since the trend of integrating information extraction and data mining is recently discussed to be promising for relation extraction problem [5], [18].

References

1. M. Völkel, M. Krötzsch, D. Vrandecic, H. Haller, and R. Studer. Semantic Wikipedia. In *Proceedings of the WWW2006*, pages 585-594, 2006.
2. T. Berners-Lee, J. Hendler, and O. Lassila. The Semantic Web. *Scientific American*, (5), 2001.
3. F. Manola and E. Miller. Resource Description Framework (RDF) primer. *W3C Recommendation*, 10 February 2004. Available at http://www.w3.org/TR/rdf-primer/.
4. T. Berners-Lee. Notation 3. *W3C Design issue*, 1998, Available at http://www.w3.org/DesignIssues/Notation3
5. A. Culotta, A. McCallum, and J. Betz. Integrating Probabilistic Extraction Models and Data Mining to Discover Relations and Patterns in Text. In *Proceedings of the HLT-NAACL-2006*, 2006.
6. S. Brin. Extracting Patterns and Relations from the World Wide Web. In *Proceedings of the 1998 International Workshop on the Web and Databases*, pages 172-183, 1998.
7. E. Agichtein and L. Gravano. Snowball: Extracting Relations from Large Plain-Text Collections. In *the* 5^{th} *ACM International Conference on Digital Libraries (ACM DL)*, 2000.
8. D. Ravichandran and E. H. Hovy. Learning Surface Text Patterns for a Question Answering System. In *Proceedings of the ACL-2002*, pages 41-47, 2002.
9. R. C. Bunescu and R. J. Mooney. Extracting Relations from Text: From Word Sequences to Dependency Paths. In *"Text Mining and Natural Language Processing"*, Anne Kao Steve Poteet (eds.), forthcoming book, 2006
10. C. Sutton and A. McCallum. An Introduction to Conditional Random Fields for Relational Learning. In *Introduction to Statistical Relational Learning*, Lise Getoor and Ben Taskar (eds.), MIT Press, 2006.
11. D. Lin. Dependency-Based Evaluation of Minipar. In *Proceedings of the Workshop on the Evaluation of Parsing Systems,* 1^{st} *International Conference on Language Resources and Evaluation*, 1998.
12. J. Pei, J. Han, B. Mortazavi-Asl, J. Wang, H. Pinto, Q. Chen, U. Dayal, and M.-C. Hsu. Mining Sequential Patterns by Pattern-Growth: The PrefixSpan Approach. *IEEE Transactions on Knowledge and Data Engineering*, 16(10), 2004.
13. C. J. Fillmore and C. F. Baker. Frame Semantics for Text Understanding. In *Proceedings of WordNet and Other Lexical Resources Workshop*, NAACL, 2001.
14. M. Palmer, D. Gildea and P. Kingsbury. The Proposition Bank: An Annotated Corpus of Semantic Roles. *Computational Linguistics*, 31(1), pages 71-106, 2005.
15. C. F. Baker, C. J. Fillmore, and J. B. Lowe. The Berkeley FrameNet Project. In *Proceedings of the COLING/ACL-98*, pages 86-90, 1998.
16. D. Gildea and D. Jurafsky. Automatic Labeling of Semantic Roles. *Computational Linguistics*, 28(3), pages 245-288, 2002.
17. P. Koomen, V. Punyakanok, D. Roth, and W. Yih. Generalized Inference with Multiple Semantic Role Labeling Systems. In *Proceedings of the CoNLL*, pages 181-184, 2005.
18. Un Yong Nahm. Text Mining with Information Extraction. *Ph.D. Thesis*. Department of Computer Sciences, University of Texas at Austin, 2004.