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. However, one of the current challenges 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 novel method to exploit syntactic and semantic information for relation extraction. We mine frequent subsequences from the path between an entity pair in the syntactic and semantic structure in order to explore key patterns reflecting the relationship between the pair. 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 analyzing language in higher level 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. Its openness leads to its exponential growth [4]. Since the encyclopedia is managed by Wikipedia Foundation, an international non-profit organization, and a great number of collaborators, 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 [1]. The explanation is that articles in Wikipedia are written in natural languages and thus they prevent machines from processing their content semantically. In order 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

[3] http://www.wikipedia.org/
[4] http://en.wikipedia.org/wiki/Wikipedia:Modelling_Wikipedia's_growth

[4]. We follow the formalism of Semantic Web, specifically N3, in which we structure Wikipedia's content as a collection of statements. Each statement consists of a subject, a predicate and an object. For example, the statement (Microsoft, *Founder*, Bill Gates) represents the knowledge of the sentence: "Bill Gates is one of the founders of the Microsoft Corporation". The statements with the use of a domain-specific ontology can then be straightforwardly transformed into RDF format that in turn serves as machine-processable knowledge base.

In this paper, we describe a novel method to deal with relation extraction problem for English version of Wikipedia encyclopedia. Our method, unlike other works, mines the key patterns from syntactic and semantic structure to measure similarity between entity pairs rather using only lexical information as in [5–8] or hard matching of dependency paths as in [9]. In details, we attempt to integrate syntactic and semantic information of text to form an unified structure. We then decompose 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 Wikipedia's nature to automatically obtain training data, which gives our system high portability for new relationships with no human labor is required.

The remainder of this paper is organized as follows. The next section describes our problem in details along with some characteristics of articles in Wikipedia. Some related works are carefully reviewed in Section 3. We explain our proposed methods for relation extraction in Section 4. Section 5 provides experiments and evaluations of our methods. Finally, we conclude and present future works in Section 6.

2 Problem statement

In this section, we define our problem along with some assumptions based on the characteristics of Wikipedia's articles.

We aim at extracting binary relations between entities from English version of Wikipedia articles. A 2-tuple (e_p, e_s) and a triple (e_p, *rel*, e_s) denote an entity pair and a binary relation respectively, where e_p and e_s are entities which may be PERSON, ORGANIZATION, LOCATION, TIME or ARTIFACT and *rel* denotes the directed relationship between e_p and e_s, which may be one of following 13 relations: CEO, FOUNDER, CHAIRMAN, COO, PRESIDENT, DIRECTOR, VICE CHAIRMAN, SPOUSE, BIRTH DATE, BIRTH PLACE, FOUNDATION, PRODUCT and LOCATION. Our system is given Wikipedia text and should return a set of triples as extracted relations.

Since Wikipedia is a free online encyclopedia, it mostly contains entries or articles that provide information for a specific entity. We follow [5] to define the entity mainly discussed in an article as *principal entity*, and other mentioned entities in the same article as *secondary entities*. We assume that interested entities in this problem should have a descriptive article in Wikipedia. Thus, no entity disambiguation and entity recognition is required in our system. 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. 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

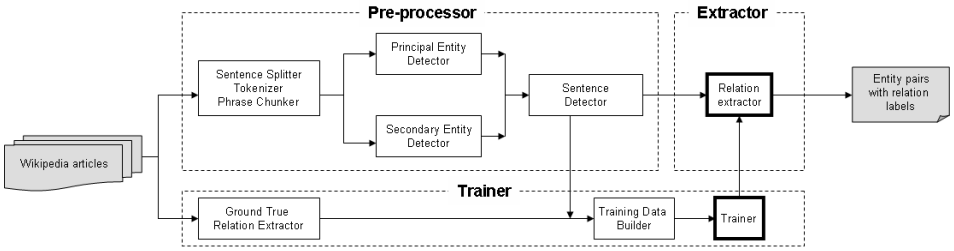

Fig. 1. System framework

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.

3 Related Works

Some important works on relation extraction by learning surface text were introduced in [6–8]. 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. Thus, they cannot handle long dependencies between words. As the result, the methods may fail in this problem since the Wikipedia source is more formal, complex but not abundant.

Recently, the authors in [9] present a kernel method to classify relationships of entity pairs by estimating similarity between dependency paths. Their method relies on an assumption that paths with different lengths tend to express different relationships. Additionally, if the paths satisfy the condition of length, the system multiplies the matching results of corresponding positions, which requires the paths to be well aligned. Their method may be more efficient if the assumption and matching condition can be relaxed. Our work attempts to overcome the problem by matching the decomposed subpaths independent of length.

Culotta et al. [5] presents a probabilistic model to integrate extraction and mining tasks performed on biographical text of Wikipedia. To avoid the suffering from the errors of the traditional pipeline, they formulate the relation extraction problem into sequence labeling problem which then is solved by Conditional Random Field. Their supervised method uses both contextual and relational information to enable the two tasks support each other to improve the whole system.

Max Völkel et al. [1] provides a tool that enables users to annotate knowledge to Wikipedia. The knowledge they support may be categories, typed links or attributes. They define typed links as links between the articles. Since we assume that an entity should have a descriptive article, typed link between articles is equivalent to relation between entities. Therefore, our work can be consider as a realization of this work when we move from manual annotation to automation.

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 both syntactic and semantic information.

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)

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 point 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 provided by OpenNLP [5] tool set. The articles are then parallelly processed to anchor all occurrences of principal entities and secondary entities. The Secondary Entity Detector simply labels appropriate surface text of the hyperlinks as secondary entities. After that, the Sentence Selector chooses only sentences which contain the principal entity and at least one secondary entity. The Trainer receives articles with HTML tags to identify summary sections and extract ground true relations annotated by human editors. Previously selected sentences that contain entity pairs from ground true relations are identified as training data. The Trainer will learn the key patterns with respect to each relation. During testing, for each sentence and an entity pair on it, the Relation Extractor will identify the descriptive label and then outputs the final results.

Principal Entity Detector From the following characteristics:
-Most of the pronouns in an article refer to the principal entity.
-The first sentence of the article is often used to briefly define the principal entity.

We use rules to identify a set of referents to the principal entity, including three types [10]: (1) pronoun ("he", "him", "they", "them"...) (2) proper noun (e.g., Bill Gates, William Henry Gates, Microsoft,...) (3) common nouns (the company, the software,...). Figure 2a shows some sample referents extracted for several articles by our rules. Supported by the nature of Wikipedia, our technique performs better than those of the coreference tools in LingPipe library [6] and in OpenNLP tool set. All the occurrences of the collected referents are labelled as principal entity.

Sentence Detector This module selects sentences that contain at least one occurrence of the principal entity and a secondary entity. Each of such pairs becomes

[5] http://opennlp.sourceforge.net/
[6] http://www.alias-i.com/lingpipe/index.html

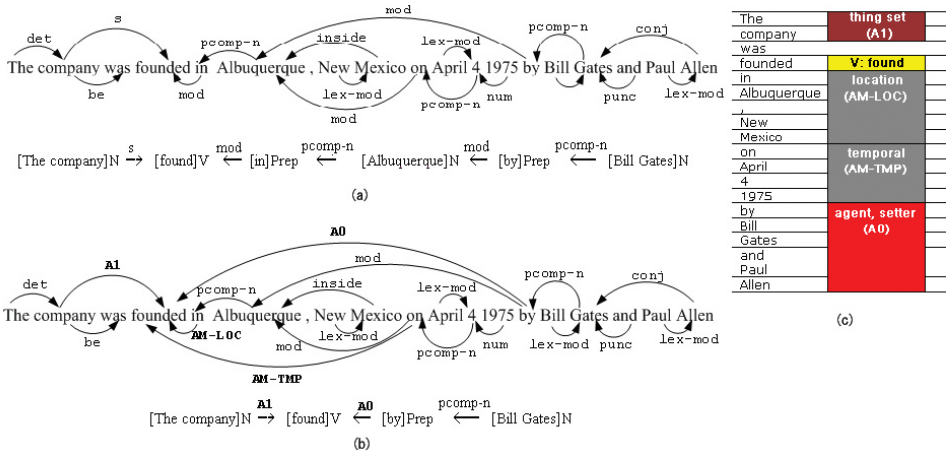

Fig. 3. Syntactic (a), semantic (c) and integrated representation (b) of a sample sentence

a relation candidate. So, there may be more than one relation candidate on a sentence.

Extract Relation from Summary Section 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 which relations (Microsoft, *Foundation*, Albuquerque), (Microsoft, *Founder*, Bill Gates)... can be extracted. We exploit such relations to create training data. From here, ground true relation refers to the relations obtained by this way.

Training Data Builder For a selected sentence and an entity pair, this module examines whether the pair is in ground true relation set or not. If yes, it attaches the relation label to the pair and create a new training sentence for the relation. For a relation 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.

One of the challenges for this problem is due to the wide variation of the surface text. However, the syntactic structures of the sentences enable us to reduce the variation. Follow the idea in [9], we assume that the shortest dependency path tracing from a principal entity through the dependency tree to a secondary entity gives a concrete syntactic structure expressing relation between the pair as shown in Figure 3a. 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 3 express FOUNDER relationship, their surface text is totally different. A closer analysis of the dependency paths between the entity pairs suggests that, if we separate the paths into tokens then they share a common segment of path *"[found]V* $\leftarrow^{mod}$ *[by]Prep* $\leftarrow^{pcomp-n}$ *N"*.

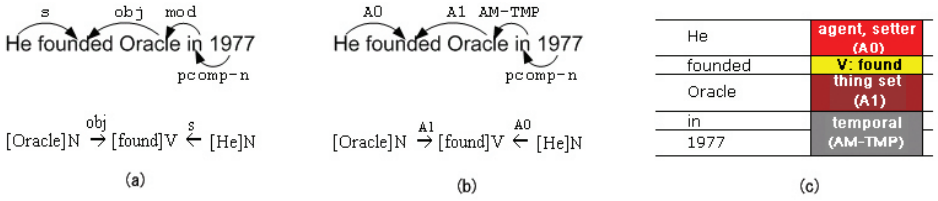

Fig. 4. Syntactic (a), semantic (c) and integrated representation (b) of another sentence

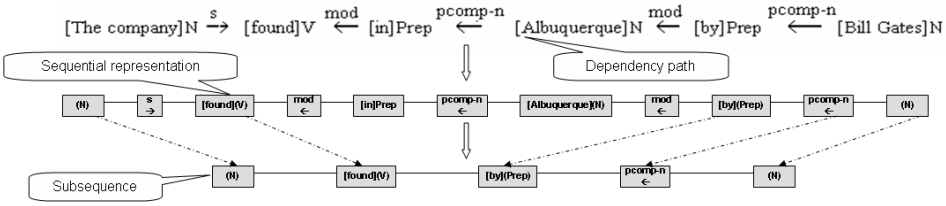

Fig. 5. Sequential representation of a dependency path

Our idea is to learn such key patterns from the dependency paths for each relationship. In particular, we firstly derive dependency trees of the training sentences by Minipar parser [11] and extract paths between entity pairs. We then transform the paths into sequences which are in turn decomposed into subsequences. From the subsequence collections of a relation r, we can identify the frequent subsequences for r. 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 r it matches, the more likely that the original sentence express relation r between the entity pair. From now, we call pattern and subsequence interchangeably.

Sequential Representation of Dependency Path A word together with its Part-Of-Speech (POS) tag will be an element of the sequence. In case of the first and the last words, only POS tag is mentioned. Similarly, a relation label and its direction will also be transformed into an element. 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. A sequence s=< $s_1 s_2 ... s_n$ >, where s_i is an itemset, is called subsequence of a sequence p=< $p_1 p_2 ... p_m$ > if there exists integers $1 <= j_1 < j_2 < ... < j_n <= m$ such that $s_1 \subseteq p_{j_1}$, ..., $s_n \subseteq p_{j_n}$. Given a sequence database, PrefixSpan will find all the subsequences appearing more frequently than a given support threshold. Our problem of learning key patterns is casted to a special case of sequence mining problem in which all itemsets contain only one item. In this research, we use the implementation tool [7] of PrefixSpan developed by Taku Kudo. From here,

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

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. The weight should incorporate the following factors:

(i) *Length of the pattern*: if two paths share a long common subpattern, 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. It is more likely that a pattern with high support should be a key pattern.

(iii) *Amount of lexical information*: although the sequences contain both words and dependency relations from the original dependency path, we found that word-based items are more important. For example, the two sentences *"He is the founder of the company"* and *"He is the director of the company"* suggest different relations due to the words *"founder"* and *"director"*.

(iv) *Number of sequence databases in which the pattern appear*: if the pattern can be found in various sequence databases, it is more likely that the pattern is common and it should not be a key pattern of any relation.

Therefore, weight of a pattern with respect to a relation r is calculated as:

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

- D_r is the sequence database of r, $support_{D_r}(p)$ is the support of p in D_r.
- *irf(p)* is Inverted Relation Frequency of p calculated by $log(\frac{|R|}{|M(p)|})$, where R is set of relations and *M(p)* is set of sequence databases in which p occurs.
- *l(p)* is length of 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 relation of the pair. We extract the dependency path P, transform P into sequential pattern and then accumulate the scores of its subsequences for each relation r:

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

- $L_r(P)$ likelihood score to say that P expresses 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:

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

4.3 Learning Patterns with Dependency Path and Semantic Role

Both of the sentences in Figure 3 and 4 express the FOUNDER relationship between a company and a person but in different surface text. The syntactic representation of the sentence in Figure 4a captures the person as the *subject* and the company as the *object*. Conversely, the company is *subject* in Figure 3a since

Table 1. The result table 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.

	RetRel	**Ret**	**Rel**	**Prec(%)**	**Rec(%)**	**F1(%)**
B0	1,962	5,975	5,975	32.84	32.84	32.84
B1	2,665	5,975	5,975	44.60	44.60	44.60
Dep	2,970	5,257	5,975	56.50	49.71	52.88
DepSRL	3,449	4,991	5,975	**69.10**	**57.72**	**62.90**

the sentence is in passive voice. Thus, no common subpattern except following single-node patterns "N", "[found]V" and "N" is found.

The above analysis suggests us to use *frame semantics* theory. A frame defines relationships between a predicate and its participants in a context, which form Predicate-Argument (PA) structure [13]. Figure 3c illustrates the PA structure of a sentence. Large corpora such as PropBank [13] and FrameNet [14] enable the process of filling PA structures with constituents from text, which is well-known as Semantic Role Labeling (SRL) task [15]. We use the SNoW-based Semantic Role Labeler [16], a state-of-the-art in SRL task which conforms the definition of PropBank and CoNLL-2005 shared task [8] 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 as follows:

(i) For each predicate P and its role R, identify headwords of the two phrases.

(ii) Place the semantic relation between the headwords into dependency tree. The relation is directed, receiving the headword of P as its head, headword of R as its tail and R as its label.

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, the only additional step is to augment dependency trees with PA structure, all the other steps are same to those of method in Section 4.2.

5 Experimental Settings

5.1 Data and baseline systems

We perform our experiments on real Wikipedia data dumped on Aug 10, 2006 [9]. For evaluation, we interest only the articles whose the summary sections provide at least one target relation listed in Section 2. Wikipedia defines templates for the summary sections. For example, some company articles may contain the template called Infobox_Company while some person articles may contain Infobox_Senetor, Infobox_Celebrity... 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.

[8] http://www.lsi.upc.edu/šrlconll/

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

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 (BOW) model, in which only words themself are concerned. 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. 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 accumulates the TFIDF scores of the words between the pair for each relation. Finally, it chooses the relation label that gives the highest score for the entity pair. The only difference between the two baseline systems is that the second one uses dependency parse tree to eliminate the irrelevant words. Only words on the dependency path between the entity pair are extracted instead of all the words between the entities in the sentence.

5.2 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 ground truth 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, meaning that ground true relations overlap with real relations in article text. However, we compare all the methods in the same setting described below. So, the results in the following section are still strongly believable and the comparison is credibly fair.

Firstly, we attempt to estimate the list of retrieved relations, relevant relations and correctly retrieved relations for our systems. Secondly, we calculate the precision and recall as usual. Let α and β be the set of ground true relations and the set of relations outputed from a 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 relation 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 the entities in α are also expressed in text and all the recognized relations between the entities 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. $|\beta'|$ is the numbers in *RetRel* column in Table 1.

•$\beta'' = \{t | t \in \beta \wedge t.rel \neq nolabel \wedge (\exists l \in \alpha : t.e_p = l.e_p \wedge t.e_s = l.e_s)\}$, 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 1.

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

Then, we calculate precision and recall:

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

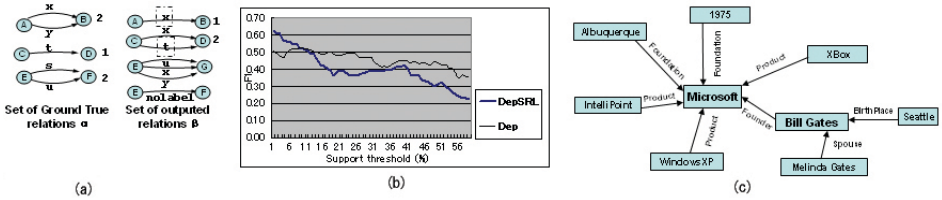

Fig. 6. (a) An example for evaluation setting (b) curves to compare the system using only syntactic information (Dep) and the system using syntactic and semantic information (DepSRL) and (c) some successfully extracted relations from our best system (DepSRL with 2% as support threshold)

The example illustrating our evaluation method is given in Figure 6a. 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. Only *(A, x, B)* and *(C, t, D)* are correctly retrieved. 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.

5.3 Results

Table 1 shows the results of four systems under the same dataset and evaluation setting. We use 5-fold cross validation to test our systems. "B0"denotes the baseline systems using trivial BOW model while the system denoted by "B1" uses BOW model together with dependency parsing to remove irrelevant words. "Dep" and "DepSRL" denote the systems mining frequent patterns from dependency paths and the system mining frequent patterns from paths in integration structure 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 table since they are far from evaluation.

As mentioned in Section 4.2, PrefixSpan algorithm accepts a support threshold as a parameter to set the minimal occurrences of the mined subsequences. Thus, "Dep" and "DepSRL" systems depend on the support threshold. The "Dep" system in Table 1 obtains the best result at 10% of 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 6b 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 may include key subsequences of the relationships. In other words, high thresholds may lose some important subsequences. 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 6c shows some relations correctly 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 work

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 patterns 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.

In the future, we plan to extend the sequence mining to subtree mining since there are some cases in which the clues for a relationship between an entity pair place outside the path between the pair.

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. T. Morton. Coreference for nlp applications. In *Proceedings of the ACL-2000*, 2000.
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. M. Palmer, D. Gildea and P. Kingsbury. The Proposition Bank: An Annotated Corpus of Semantic Roles. *Computational Linguistics*, 31(1), pages 71-106, 2005.
14. 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.
15. D. Gildea and D. Jurafsky. Automatic Labeling of Semantic Roles. *Computational Linguistics*, 28(3), pages 245-288, 2002.
16. 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.