Subtree Mining for Relation Extraction from Wikipedia

Abstract

In this study, we address the problem of wide variation of surface text in relation extraction by mining frequent subtrees for each relationship from syntactic structure. With the goal to extract knowledge from Wikipedia source, we also make use of Wikipedia's web structure to anchor the appearance of entities in the articles using neither Named Entity Recognizer (NER) nor coreference resolution tool, and to classify entity type which is essential for the whole task. We evaluate our method on manually annotated data from actual Wikipedia articles.

1 Introduction

The emergence of the WWW yields the dramatic increase of textual information. There is a great demand for organizing such textual data into structure to support machine-processable. To that end, the goal of *relation extraction* techniques is to locate interesting entities and identify relations between them (Culotta and Sorensen, 2004; Zhou et al., 2005).

This study is intended to deal with the problem of converting Wikipedia's English version (http://en.wikipedia.org), a free encyclopedia on the web, into structure by using relation extraction because of its size and its reliability. The term *wiki* indicates that information can be freely appended to the online encyclopedia by anyone who can access the site, which has engendered the exponential growth of the encyclopedia[1]. Furthermore, because the encyclopedia is managed by the Wikipedia Foundation, an international non-profit organization, and because numerous collaborators in the world participate under some international projects, its articles are edited and developed continuously. This means that its contents is quite reliable despite its openness.

Relations used to structure Wikipedia in this task are defined in form of 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}. Current experiment limits entities and relations to a reasonable size in that an entity is classifiable as one of seven types: *person*, *organization*, *location*, *artifact*, *year*, *month* or *date*; and a relation can be one of 13 types: *founder*, *chairman*, *CEO*, *COO* (Chief Operating Officer), *president*, *director*, *vice chairman*, *spouse*, *birth date*, *birth place*, *foundation*, *product* and *location*. For example, relation (Microsoft Corp., *founder*, Bill Gates) should be extracted from sentence *"Bill Gates is a founder of Microsoft Corp."*.

We propose a supervised learning method based on analyses of the syntactic structure of text to address this problem. Unlike the web, Wikipedia articles contain few pieces of text that are relevant to each relationship between an entity pair. In other words, Wikipedia contents are not so abundant. Furthermore, Wikipedia text is believed to be *clean* compared to that of the web overall. Those assumptions enable us to use deep analyzing techniques, which is usually infeasible for ordinary web pages.

[1] http://en.wikipedia.org/wiki/Wikipedia:Modelling_Wikipedia's_growth

Long dependencies between constituents (words or group of words which form a single unit in a sentence) in text are handled to improve recall. Put differently, analyzing the text at a syntactic level allows reduction of the variation of superficial text, which subsequently enables machines to recognize entity relations more accurately.

Although previous works (Bunescu and Mooney, 2006; Cui et al., 2005) attempt to analyze the relation path between entity pair, which is extracted from syntactic structure of text, our method analyzes a subtree derived from the structure. Such a subtree contains more evidence of the entities' inter-relation than the path in some cases. We propose a new feature obtained using a subtree-mining technique.

In addition to analysis of the Wikipedia text, we also make use of the characteristics of Wikipedia articles to narrow down the list of relations which might pertain between an entity pair. Specifically, the category hierarchy of Wikipedia is the main feature in our method to classify the entity type.

The contributions of our research can be summarized as follows:

- This research shows a good application of linguistic techniques because Wikipedia is a well-formed and it conveys the latest information for large Web users.
- This research seeks the possibility to use the knowledge extracted from the web for linguistic technologies such as summarization, word-sense disambiguation, and question answering.

The remainder of this paper is organized as follows. The next section describes some related works. Section 3 presents an analysis of the characteristics of Wikipedia articles that are useful for this research. We explain our proposed methods for relation extraction and entity classification in Section 4. Section 5 describes experimental results and evaluations of our methods. Finally, we conclude this paper in Section 6.

2 Related Works

Some earlier works on relation extraction using learning surface text have been introduced into the literature (e.g., (Brin, 1998; Agichtein and Gravano, 2000; Ravichandran and Hovy, 2002)). The authors conducted experiments on web data that are so abundant that they enable their systems to obtain easy patterns. The systems then learn such patterns based mostly on lexical information. Consequently, they cannot cope with long dependencies between words. As a result, the methods might fail in this problem because the Wikipedia source data are more formal and complex, but not abundant.

There are many interests on kernel method for relation extraction. The main idea of this approach is to measure the similarity between instances of relation. (Zelenko et al., 2003; Culotta and Sorensen, 2004) define tree kernels that measure the similarity between parse trees containing the mentions of entities. (Zhang et al., 2006) proposes a composite kernel from the two seperate kernels: one reflects the entities' features and one reflects syntactic relations between the entities. Bunescu et al. (Bunescu and Mooney, 2006) relies on an assumption that dependency paths of relation instances with different lengths tend to express different relationships. If the paths satisfy the condition of length, their kernel multiplies the matching results of corresponding positions, otherwise it will be zero. The kernel requires the paths to be well aligned. It might be more efficient if such a hard matching condition were relaxed.

To our knowledge, only one recent work has attempted relation extraction on Wikipedia: Culotta et al. (2006) presents a probabilistic model to integrate extraction and mining tasks performed on biographical text from Wikipedia. They formulate the relation extraction problem into a sequence labeling problem, which then is solved using Conditional Random Field to avoid errors of the traditional pipeline, including the Named Entity Recognizer (NER). Their supervised method uses both contextual and relational information to enable the two tasks to be mutually supportive and thereby improve the entire system. Our work resembles that effort, but the present work is motivated more by the Semantic Web vision: we intend to convert Wikipedia texts into machine-processable knowledge for Semantic Web using NLP techniques. Therefore, we also use some special characteristics that are intrinsic to Wikipedia.

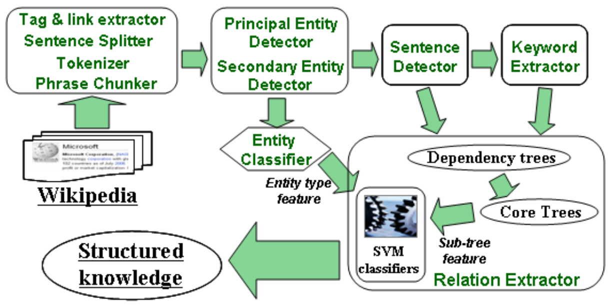


Figure 1: System framework

3 Wikipedia's Article Characteristics

Because Wikipedia has an *encyclopedic style*, it mainly contains entries or articles, each of which provides information for a specific entity and further mentions other entities related to it. (Culotta et al., 2006) respectively defines the entities as *principal entity* and *secondary entity*. In this research, we predict only relationships between the principal entity and each mentioned secondary entity that contains a link to its descriptive article.

We put the following assumptions in this study: a relationship can be expressed completely in one sentence. Furthermore, a relationship between an entity pair might be expressed with the implication of the principal entity in some sentences. Thus, for an article, only the sentences that contain at least one secondary entity are necessarily analyzed.

An interesting characteristic of Wikipedia is the existing category hierarchy that is used to group articles according to their content. Additionally, those articles for famous entities provide summary sections on their right side, which are created by human editors. Finally, the first sentence of an article often defines the principal entity. We exploit such characteristics in this research.

4 Proposed Method

Figure 1 delineates our framework for relation extraction. First, articles are processed to remove HTML tags and to extract hyperlinks that point to other Wikipedia articles. Text is then submitted to a pipeline including a *Sentence Splitter*, a *Tokenizer* and a *Phrase Chunker* supplied by the OpenNLP [2] tool set. The instances of the principal entity and secondary entities are then anchored in the articles. The *Secondary Entity Detector* simply labels the appropriate surface texts of the hyperlinks to other Wikipedia articles, which are proper nouns as secondary entities. The *Principal Entity Detector* will be explained in the following subsection.

After the entities are anchored, sentences that include at least one mention of secondary entities will be selected by a *Sentence Detector*. Each mention of the secondary entities should be analyzed to identify the relation between the underlying entity and the principal entity. Secondary entities are always explicit, although the principal entity is sometimes implicit in sentences containing no mention.

Keywords that provide clues for each relation label will be identified by a *Keyword Extractor*. Similarly, an *Entity Classifier* module will classify the entities into types to limit the available relations for entity pairs. The *Relation Extractor* will extract *subtree feature* from a pair of the principal entity and a mention of secondary entity. It then incorporates *subtree feature* together with *entity type feature* into a feature vector and classifies relations of the entity pair using *SVM-based classifiers*.

4.1 Principal Entity Detector

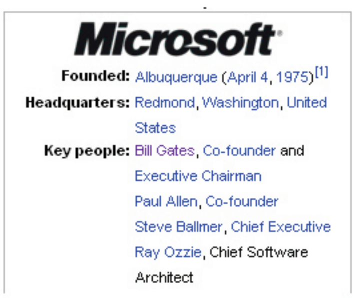
This module detects all instances of the principal entities in an article. The function of this module is classifiable as *coreference resolution* for noun phrases (Soon et al., 2001; Ng and Cardie, 2002; Morton, 2000), in which *referring expressions*, noun phrases that refer to the principal entity, are identified. All occurrences of identified referring expressions are labeled as mentions of the principal entity. We adopt (Morton, 2000) to classify the expressions into three types: (1) personal pronoun (2) proper noun (3) common nouns. Based on chunking information, we propose a simple technique to identify a set of referring expressions, denoted as F:

(i) Start with $F = \{\}$.

(ii) Select the first two chunks for F: the proper chunk (chunk with at least a proper noun) of the *article title* and the first proper chunk in the *first sentence* of the article, if any. If F is still empty, stop.

(iii) For each remaining proper chunk p in the article, if p is derived from any expressions selected in

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


Microsoft

Founded: Albuquerque (April 4, 1975)[1]

Headquarters: Redmond, Washington, United States

Key people: Bill Gates, Co-founder and Executive Chairman
Paul Allen, Co-founder
Steve Ballmer, Chief Executive
Ray Ozzie, Chief Software Architect

Figure 2: The summary section in Wikipedia's Microsoft Corp. article

Table 1: Sample extracted referring expressions

Article	Referring expressions	Step
Bill Gates	[NP Bill/NNP Gates/NNP]	(ii)
	[NP William/NNP H./NNP Gates/NNP]	(ii)
	[NP Gates/NNP]	(iii)
	[NP The/DT Gates/NNP]	(iii)
	[NP he/PRP]	(iv)
	[NP him/PRP]	(iv)
Microsoft	[NP Microsoft/NNP]	(ii)
	[NP The/DT Microsoft/NNP Corporation/NNP]	(ii)
	[NP that/DT Microsoft/NNP]	(iii)
	[NP It/PRP]	(iv)
	[NP the/DT company/NN]	(v)
Microsoft Windows	[NP Microsoft/NNP Windows/NNP]	(ii)
	[NP Microsoft/NNP]	(iii)
	[NP Windows/NNP]	(iii)
	[NP the/DT Windows/NNP]	(iii)
	[NP it/PRP]	(iv)

(ii), then $F \leftarrow p$. Proper chunk p_1 is derived from proper chunk p_2 if all its proper nouns appear in p_2.

(iv) In the article, select *c* as the most frequent *subjective pronouns*, find *c'* as its equivalent *objective pronoun* and add them to *F*.

(v) For each chunk p with the pattern [DT $N_1 \ \ldots \ N_k$] where DT is a *determiner* and N_k's are a *common nouns*, if p appears more frequently than all the selected pronouns in (iv), then $F \leftarrow p$.

Table 1 shows some sample referring expressions extracted by the above technique. The third column indicates in which step the expressions are selected. Supported by the nature of Wikipedia, our technique provides better results than those of the coreference tool in LingPipe library [3] and OpenNLP tool set.

4.2 Entity Classifier

Entity type is very useful for relation extraction. For instance, the relation label between a *person* and an *organization* should be *founder*, *chairman*, etc., but cannot be *spouse*, *product*, etc. In order to classify entities, we first identify *year*, *month* and *date* entities by directly examining their surface text. Types of other entities are identified by classifying their corresponding articles. We develop one SVM-based classifier for each remaining type using one-against-all strategy. We represent an article in the form of a feature vector and use the following features: **category feature** (categories collected when tracing from the article up to k levels of its category structure), **pronoun feature** (the most frequent *subjective pronoun* in the article) and **singular noun feature** (singular nouns of the first sentence of the article).

An appropriate number of slots in feature vector is dedicated to each feature. Each slot corresponds to a value of the features. We assign value 1 to the slots corresponding to the actual values of the features. The pronoun feature might receive only one value, whereas the category feature and singular noun feature might receive multiple values.

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

Table 2: List of relations and their keywords

Relation	Keywords
CEO	CEO, chief, executive, officer
Chairmans	chairman
COO	coo, chief, operating, officer
Director	director
Founder	found, founder, founded, establish, form, foundation, open
President	president
Vice chairman	vice, chairman
Birth date	born, bear, birth, birthday
Birth place	born, bear
Foundation	found, establish, form, founded, open, create, formed, established, foundation, founding, cofounder, founder
Location	headquartered, based, locate, headquarter, base, location, situate, located
Product	product, include, release, produce, service, operate, provide, market, manage, development, focus, manufacture, provider, launch, make, sell, introduce, producer, supplier, possess, retailer, design, involve, production, offering, serve, sale, supply
Spouse	marry, wife, married, husband, marriage

4.3 Keyword Extractor

Our hypothesis in this research is that there exist some keywords that provide clues for the relationship between a pair. For example, to express the *founder* relation, a sentence should contain one keyword such as: *found*, *founder*, *founded*, *co-founders*, or *establish*, etc. We identify such keywords by using a semi-automatic method. First, we automatically extract some true relations from summary sections of Wikipedia articles. Some articles about famous and important entities contain a summary section as shown in Figure 2. Then, we map entities in such true relations to those in sentences to build sample sentences for each relationship . *Tf-idf*

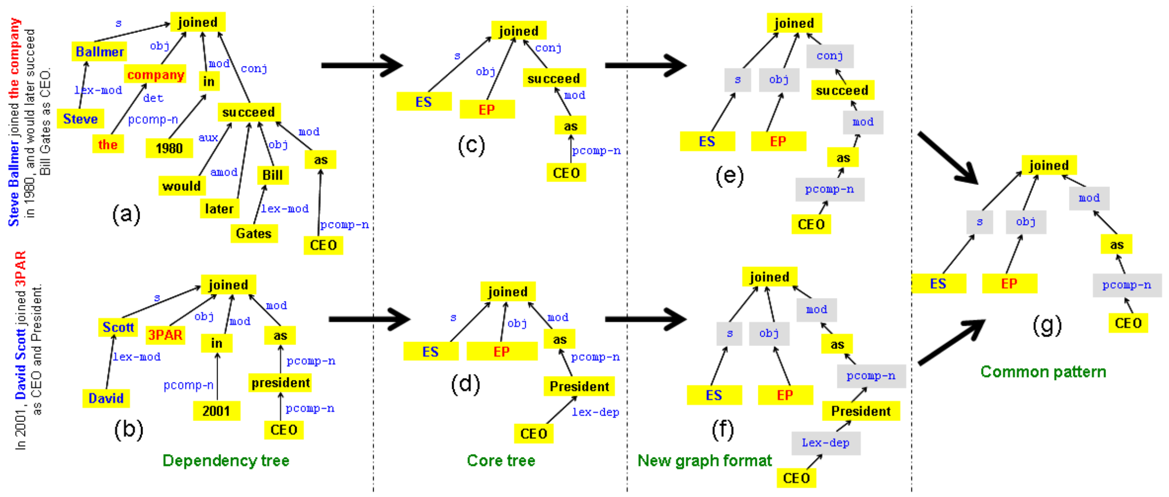

Steve Ballmer joined the company in 1980, and would later succeed Bill Gates as CEO.
In 2001, David Scott joined 3PAR as CEO and President.
(a)
(b)
(c)
(d)
(e)
(f)
(g)
Dependency tree
Core tree
New graph format
Common pattern

Figure 3: Dependency trees in (a) & (b); core trees with respect to *CEO* relationship in (c) & (d); new representation of the core trees in (e) & (f); common subtree in (g). The red phrase *EP* denotes the principal entity; the blue phrase *ES* denotes the secondary entity.

model is exploited to measure the relevance of words to each relationship for those on the dependency path between the entity pair. Finally, we choose the keywords manually from lists of candidates ranked by relevance score with respect to each relation. Table 2 shows our result selected from ranked lists of total 35,820 keyword candidates using only one hour of human labor.

4.4 Subtree Feature from Dependency Path

In this section, we will describe how to obtain efficient features for extracting relation using subtree mining. One challenge for this problem is posed by the wide variation of surface text styles. However, thanks to syntactic analysis, we can explore relations between words in a sentence even if they are separated by a long distance. In other words, the syntactic analysis of sentences enables us to reduce the variation of the sentences. In this study, we extract relations between entities by analyzing a dependency graph of sentences. (Bunescu and Mooney, 2006) investigated the sentences in an Automated Content Extraction [4] (ACE) newspaper corpus and suggested that clues for the relationship between two entities in a sentence be placed on the shortest dependency path between the entities.

Some analyses suggest that Wikipedia sentences in which one entity of the pair is implied might counteract the hypothesis. For example, although the sentence shown in Fig. 3a shows the *CEO* relationship between *Steve Ballmer* and *the company*, the dependency path *"[the company]N* $\rightarrow^{obj}$ *[joined]V* $\leftarrow^{s}$ *[Steve Ballmer]N"* between the entities shows no clue to the relationship.

To investigate whether a relationship r is held between the entities or not, our novel idea is to expand such a dependency path to a tree that contains as many clues for r as possible and then analyze the tree. We expand the path by integrating more paths between the secondary entity and the keywords of r, as shown in Section 4.3. The expanded tree is defined as *core tree* of r because it attempts to capture the clues for r. Steps to extract the core tree C of a relationship r from a sentence s are described as follows:

(i) Initialize the core tree C as blank.

(ii) Derive the dependency tree D from s.

(iii) Label the group of nodes corresponding to words of secondary entity by an ES node in D.

(iv) If the principal entity appears in s, apply (iii) to replace principal entity with EP. Then extract P_0 as shortest path from ES to EP in D and add $P_0 \rightarrow C$.

(v) For each keyword w of r, extract P_w as the shortest path from ES to node of w and add $P_w \rightarrow C$.

Figures 3c & 3d present exemplary core trees of *CEO* relationship derived from the dependency trees

[4] http://www.nist.gov/speech/tests/ace/

in Figures 3a & 3b. To analyze both words and relations of a core tree uniformly, we transform it into a *new graph format* (Figures 3e & 3f) in which core tree words and relations are also represented as graph nodes.

We define a basic element of a relationship r as a key pattern that commonly appears in various core trees of r. As an example, the core trees in Figures 3e & 3f share a common pattern in Figure 3g. Intuitively, this subtree shares the core trees of sentences that express the idea of *"joined the company as CEO"* or *"joined the company and do something as CEO"*.

We denote $T = (V, E)$ as a directed tree, in which V is a set of nodes and E is a set of directed edges. Node y is an ancestor of node x, denoted by $x \prec y$, if $(x, y) \in E$ or $\exists i_1, ..., i_k$ ($k \in \mathbb{N}$ and $k \geq 1$) such that $(x, i_1), (i_1, i_2), ..., (i_{k-1}, i_k), (i_k, y) \in E$. We define that a tree $S = (V_S, E_S)$ is a subtree of T if and only if: (i) $V_S \subset V$, and (ii) $\forall(x, y) \in E_S$, we have $x \prec y$ in T.

We use a subtree as a feature for relation extraction. From a set of training sentences with respect to a relationship r, we derive the core trees. Therefore, it is necessary to generate all subtrees from the set of core trees to form the feature space. A frequent tree-mining algorithm (Zaki, 2002) is used to generate subtrees from a set of core trees. A minimum support parameter is used in this algorithm to allow filtering: only the subtrees that appear more frequently than the minimum support are outputed. Assume that a relation has an appropriate core tree given a relationship, each mined subtree corresponds to a subtree feature value of the relation instance with respect to the relationship.

4.5 Supervised Learning for Relation Extraction

We formulate our problem of relation classification into a multiclass and multi-label problem in which one SVM-based classifier is dedicated for a relation.

Sentence Selector is run on the sentences of training articles to select sentences containing at least one mention of a secondary entity. For a relation r, we manually select a set of positive instances, Pos_r, from relation candidates that express the relationship r between the entity pair. We also choose negative instances in which no target relation is expressed. During training for the classifier of relation r, only instances in Pos_r serve as positive samples, all the others are used as negative samples.

We represent each pair of a principal entity and secondary entity in a sentence with respect to a relation r as a feature vector receiving values 0 and 1 . Feature vectors are created from the type of principal entity (first eight slots), type of secondary entity (next eight slots), and the mined subtree of the sentence (the remaining slots). The number of slots for a subtree feature depends on the relation.

5 Experiments and Evaluations

In this experiment, 5,975 articles are selected, in which 45 articles are for testing and 5,930 articles for training. We apply the framework in Figure 1 on the training articles to extract keywords and select relation candidates. Subsequently, 3,833 positive instances and 805 negative instances from the candidates are annotated to train the *Relation Extractor*. Among 39,467 entities collected from all principal and secondary entities, we randomly select 3,300 entities and manually annotate their types to develop the *Entity Classifier*. Finally, 3,100 entities are used for training and 200 entities are used for testing.

We develop two baseline systems to evaluate our method, which use bag-of-words model. The second system (B_1 in Table 3) works like the Keyword Extractor on training instances in that it calculates *tf-idf* scores for words on the dependency path between the entities with respect to each relation. During testing, it accumulates the *tf-idf* scores of the words on the path and chooses the relation label that gives the highest score for the entity pair. The only difference between the two baseline systems is that the first one (B_0 in Table 3) does not use a dependency parse tree; instead, it focuses on all the words between the entities in sentence text. In other words, we attempt to evaluate the contribution of syntactic information in this problem.

In our experiments, dependency graphs of sentences are obtained using the Minipar parser (Lin, 1998). Furthermore, all the classifiers are trained using SVM Light (Joachims, 1999) with 2^{nd}- order polynomial kernel function. The frequent tree miner FREQT [5] is used to mine dependency subtrees.

[5]http://chasen.org/~taku/software/freqt/

Table 3: Compare our proposed system and baselines (B_0 & B_1: baselines; $Deptree_0$: use only Entity type feature; $Deptree_1$: use only Subtree feature; $Deptree_2$: use both Subtree feature and Entity type feature)

	Precision(%)	**Recall(%)**	**F1(%)**
B_0	8.70	22.26	12.51
B_1	9.88	25.31	14.21
$DepTree_0$	16.73	41.79	23.89
$DepTree_1$	24.17	25.57	24.85
$DepTree_2$	**29.07**	**53.86**	**37.76**

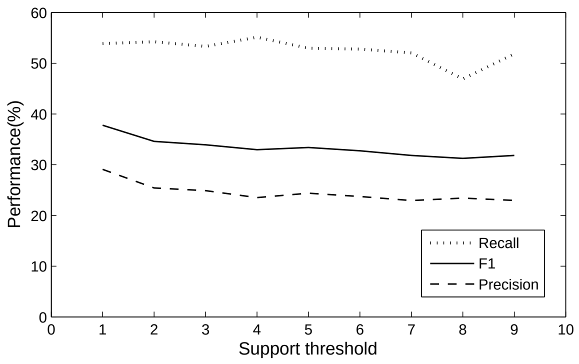


Figure 4: Performance of the system with various support thresholds

Table 4: Result of *Entity Classifier* to evaluate each feature and various values of K parameter (levels of exploited category structure)

	Depth *K*	**Accuracy(%)**
Without *pronoun feature*		80.5
Without *singular noun feature*		80.5
Without *category feature*		63.0
All features	1	64.0
	2	69.5
	3	81.0
	4	**81.5**
	5	79.5
	6	77.5
	7	77.0
	8	78.0
	9	75.0
	10	74.5

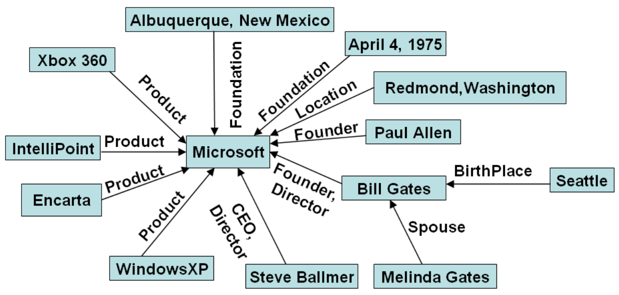


Figure 5: Some extracted relations by our system

On the basis of preliminary experiments, we report the performance of our system compared with those of baseline systems in Table 3. The result shows that our proposed method in $Deptree_2$ gives a substantial improvement over the baselines B_0 and B_1. Besides, using a few information from dependency tree in B_1 slightly improves the performance in B_0. The system $Deptree_0$ and $Deptree_1$ individually evaluate the performance of the entity type feature and subtree feature for relation classification. The best result is produced when both of the features are combined. It is clear that both of the features are essential for this task. For the result shown in $Deptree_2$, although the recall is quite adequate, precision is low. Data analysis reveals that our negative training set lacks useful negative samples to determine appropriate boundaries for the classifiers.

We also report our system's performance when varying the minimum support parameter of the frequent tree-mining algorithm in Figure 4. Although the high values of minimum support might remove the subtree features that occur infrequently in the training set, they also remove some useful features. Thus, the best system is obtained when the minimum support is set to 1, meaning that all the subtrees that are mined from training data are used as features for extracting relations.

Table 4 shows the importance of each feature used in *Entity Classifier*. The performance is slightly degraded when we discard either *pronoun feature* or *singular noun feature*. However, removing *category feature* considerably reduces the performance. The classifier works best when we incorporate all the features and trace four levels on category hierarchy. The interesting fact is that the quite high performance of this module allows Wikipedia to be used as an external knowledge source for Named Entity

Recognition. Particularly, Wikipedia currently supports a search service that returns some most appropriate articles for a given name or phrase. The search result can be passed to our module to return the entity type for the input.

Figure 5 shows some relations extracted by the system. The readers can access our research page [6] for more details of our preliminary results.

6 Conclusions and Future Works

We have presented a method to extract relations between entities from Wikipedia articles by incorporating information from the Wikipedia structure and by the analysis of Wikipedia text. The key features of our method include: (1) an algorithm to build the core syntactic tree that reflects the relation between a given entity pair more accurately; (2) the use of a tree-mining algorithm to identify the basic elements of syntactic structure of sentences for relationships; (3) method to make use of the nature of Wikipedia for entity allocation and entity classification.

As a future work, we will increase the amount of training data to improve the poor precision obtained in this study. Furthermore, we plan to incorporate information from multiple articles to predict a single relation. For instance, clues from both the *Microsoft* article and the *Bill Gates* article give stronger evidence supporting for the *founder* relationship between the entities.

Additionally, we intend to make use of more Wikipedia features, such as the link structure or various list-like articles of Wikipedia. Aside from the summary sections described in Section 4.3, some articles provide information in the form of a list. Although they cannot be processed directly by machines, they are quite structured, in contrast to free-form text.

[6]http://hitheone.dyndns.org/WikiRelExtraction.html

References

E. Agichtein and L. Gravano. 2000. Snowball: Extracting relations from large plain-text collections. In *the 5^{th} ACM International Conference on Digital Libraries.*

S. Brin. 1998. Extracting patterns and relations from the world wide web. In *Proceedings of the 1998 International Workshop on the Web and Databases*, pages 172–183.

R.C. Bunescu and R.J. Mooney. 2006. Extracting relations from text: From word sequences to dependency paths. In Anne Kao and Steve Poteet, editors, *Text Mining and Natural Language Processing.*

H. Cui, R. Sun, K. Li, M.-Y. Kan, and T.-S. Chua. 2005. Question answering passage retrieval using dependency relations. In *Proceedings of SIGIR.*

A. Culotta and J. Sorensen. 2004. Dependency tree kernels for relation extraction. In *Proceedings of the ACL-2004.*

A. Culotta, A. McCallum, and J. Betz. 2006. Integrating probabilistic extraction models and data mining to discover relations and patterns in text. In *Proceedings of the HLT-NAACL-2006.*

T. Joachims. 1999. Making large-scale svm learning practical. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*. MIT-Press.

D. Lin. 1998. 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.*

T. Morton. 2000. Coreference for nlp applications. In *Proceedings of the ACL-2000.*

V. Ng and C. Cardie. 2002. Identifying anaphoric and non-anaphoric noun phrases to improve coreference resolution. In *Proceedings of the 19th International Conference on Computational Linguistics (COLING-02).*

D. Ravichandran and E. Hovy. 2002. Learning surface text patterns for a question answering system. In *Proceedings of the ACL-2002*, pages 41–47.

W.M. Soon, D.C.Y. Lim, and H.T. Ng. 2001. A machine learning approach to coreference resolution of noun phrases. *Computational Linguistics*, 27.

M.J. Zaki. 2002. Efficiently mining frequent trees in a forest. In *Proceedings of 8th ACM SIGKDD.*

D. Zelenko, C. Aone, and A. Richardella. 2003. Kernel methods for relation extraction. *Journal of Machine Learning Research*, 3:1083–1106.

M. Zhang, J. Zhang, J. Su, and G. Zhou. 2006. A composite kernel to extract relations between entities with both flat and structured features. In *Proceedings of the ACL-2006.*

G. Zhou, J. Su, J. Zhang, and M. Zhang. 2005. Exploring various knowledge in relation extraction. In *Proceedings of the ACL-2005.*