Subtree Mining for Relation Extraction from Wikipedia

Dat P.T. Nguyen
University of Tokyo
7-3-1 Hongo, Bunkyo-ku
Tokyo 113-8656, Japan
nptdat@mi.ci.i.u-tokyo.ac.jp

Yutaka Matsuo
National Institute of Advanced
Industrial Science and Technology
Sotokanda 1-18-13
Tokyo 101-0021, Japan
y.matsuo@aist.go.jp

Mitsuru Ishizuka
University of Tokyo
7-3-1 Hongo, Bunkyo-ku
Tokyo 113-8656, Japan
ishizuka@i.u-tokyo.ac.jp

Abstract

In this study, we address the problem of extracting relations between entities from Wikipedia's English articles. Our proposed method first anchors the appearance of entities in Wikipedia's articles using neither Named Entity Recognizer (NER) nor coreference resolution tool. It then classifies the relationships between entity pairs using SVM with features extracted from the web structure and subtrees mined from the syntactic structure of text. We evaluate our method on manually annotated data from actual Wikipedia articles.

1 Introduction

Wikipedia (www.wikipedia.org) has emerged as the world's largest online encyclopedia. Because the encyclopedia is managed by the Wikipedia Foundation, and because numerous collaborators in the world continuously develop and edit its articles, its contents are believed to be quite reliable despite its openness.

This study is intended to deal with the problem of extracting binary relations between entity pairs from Wikipedia's English version. A binary relation is defined as a triple (e_p, *rel*, e_s) in which e_p and e_s are entities and *rel* indicates a directed relationship of e_p and e_s. Current experiment limits entities and relations to a reasonable size in that an entity is classifiable as *person*, *organization*, *location*, *artifact*, *year*, *month* or *date*; and a relation can be *founder*, *chairman*, *CEO*, *COO*, *president*, *director*, *vice chairman*, *spouse*, *birth date*, *birth place*, *foundation*, *product* and *location*.

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 of Wikipedia. Some other works (Brin, 1998; Agichtein and Gravano, 2000; Ravichandran and Hovy, 2002) rely on the abundance of web data to obtain easy patterns and learn such patterns based mostly on lexical information. Rather than analyzing dependency path between entity pair proposed in (Bunescu and Mooney, 2006; Cui et al., 2005), our method analyzes a subtree derived from the dependency structure. Such subtree contains more evidence of the entities' inter-relation than the path in some cases. We propose a new feature obtained from the subtree by using a subtree-mining technique.

In addition, we also make use of the characteristics of Wikipedia to allocate the mentions of entities and further identify their types to help the relation extraction process.

2 Wikipedia's Article Characteristics

Due to the *encyclopedic style*, each Wikipedia article mainly provides information for a specific entity and further mentions other entities related to it. Culotta et al. (2006) defines the entities as *principal entity* and *secondary entity* respectively. We predict only relationships between the principal entity and each mentioned secondary entity that contains a link to its descriptive article.

We put some 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 cases. Thus, for an article, only sentences containing at least a secondary entity are necessarily analyzed.

An interesting characteristic of Wikipedia is the category hierarchy that is used to classify 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.

3 Proposed Method

Figure 1 delineates our framework for relation extraction. First, Wikipedia articles are processed to remove HTML tags and to extract hyperlinks that point to other Wikipedia articles. Raw text is submitted to a pipeline including a *Sentence Splitter*, a *Tokenizer* and a *Phrase Chunker* supplied by the OpenNLP [1] 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

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

Proceedings of NAACL HLT 2007, Companion Volume, pages 125–128,
Rochester, NY, April 2007.

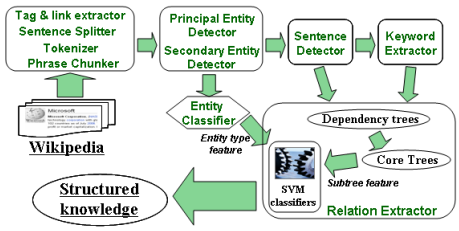

Figure 1: System framework

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 is considered as a relation candidate 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*. Parallely, an *Entity Classifier* module classifies the entities into types. The *Relation Extractor* extracts *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*.

3.1 Principal Entity Detector

This module detects all *referring expressions* of the principal entity in an article. All occurrences of identified 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 of the principal entity, denoted as F:

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

(ii) Select the first two chunks for F: the *proper chunk* (nounphase with at least one 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 (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 ... 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: 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)

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

Table 1 shows some extracted referring expressions. 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 [2] and OpenNLP tool set.

3.2 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. 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 the following features: **category feature** (categories collected when tracing from the article upto k level 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).

3.3 Keyword Extractor

Our hypothesis in this research is that there exist some keywords that provide clues for the relationship between

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

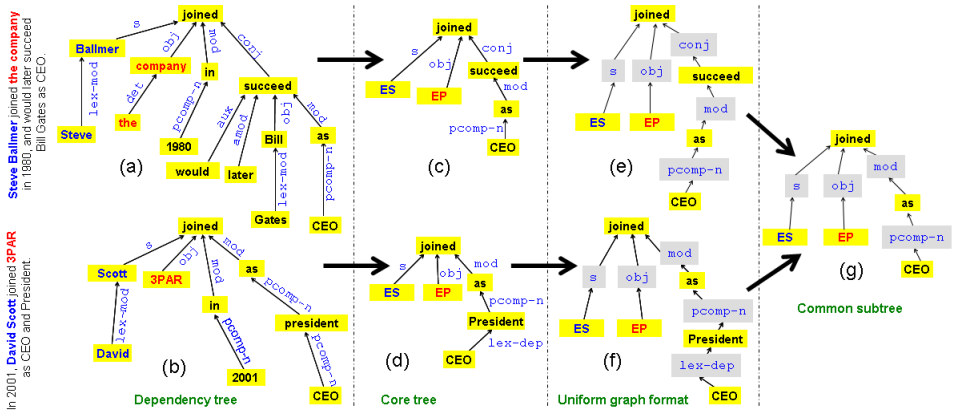

Figure 2: Dependency trees in (a) & (b); core trees with respect to *CEO* relationship in (c) & (d); new representation of the core trees in (e) & (f); common subtree in (g). The red phrase *EP* denotes the principal entity; the blue phrase *ES* denotes the secondary entity.

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. Then, we map entities in such relations to those in sentences to build sample sentences for each relationship . *Tf-idf* 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.

3.4 Subtree Feature from Dependency Path

In this subsection, we will describe how to obtain efficient features for extracting relation using subtree mining. We extend the idea of Bunescu et al. (Bunescu and Mooney, 2006) suggesting the analysis of dependency path between the entities for extracting relation, in that paths between the secondary entity and the keywords of r will be added to the dependency path between the entities to create a tree. 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 2c & 2d present exemplary core trees of *CEO* relationship derived from the dependency trees in Figures 2a & 2b. To analyze both words and relations of a core tree uniformly, we transform it into a *uniform graph format* (Figures 2e & 2f) 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 2e & 2f share a common pattern in Figure 2g. 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. A frequent tree-mining algorithm (Zaki, 2002) is used to generate subtrees from that set of core trees to form the feature space. Each mined subtree corresponds to a value of the feature.

4 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 (each contains at least one relation) and 805 negative instances (the ones containing no relation) from the candidates are annotated to train the *Relation Extractor*. Among 39,467

Table 3: Compare our proposed system and baselines

	Precision(%)	Recall(%)	F1(%)
B0	8.70	22.26	12.51
B1	9.88	25.31	14.21
DepTree	29.07	53.86	37.76

Table 4: Result of *Entity Classifier* with various levels (k value) of exploited category structure

Depth k(%)	Accuracy(%)
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

entities collected from all principal and secondary entities, we randomly select 3,300 entities and manually annotate their types for the *Entity Classifier*. Finally, we use 3,100 entities for training and 200 entities for testing.

We develop two baseline systems to evaluate our method, which use bag-of-words model. The second system (B1 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 *tf-idf* scores of 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 (B0 in Table 3) focuses on all the words between the entities in sentence text, not dependency path.

In our experiments, dependency graphs are obtained by Minipar parser (Lin, 1998), classifiers are trained by SVM Light (Joachims, 1999) with 2^{nd}- order polynomial kernel, subtrees are mined by FREQT [3] tree miner.

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 gives a substantial improvement over the baselines. Although the recall is quite adequate, precision is low. Data analysis reveals that although the mined subtrees capture key features for relationships, they also generate many irrelevant features which degrade the performance. It is necessary to carry out feature selection step for subtree feature. One more reason of the poor precision is that our system suffers from the error accumulation in a long pipeline of *entity detection*, *entity classification*, *dependency parsing* and *relation classification*.

Table 4 shows the effectiveness of different values of k parameter in *Entity Classifier*. The classifier works best when we trace *four levels* on category system. An interesting fact is that Wikipedia can be used as an external knowledge source for Named Entity Recognition.

[3] http://chasen.org/taku/software/freqt/

5 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.

References

E. Agichtein and L. Gravano. 2000. Snowball: Extracting relations from large plain-text collections. In *the 5th 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, 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,* 1st *International Conference on Language Resources and Evaluation*.

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

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

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