Relation Classification for Semantic Structure Annotation of Text

Yulan Yan
The University of Tokyo
yulan@mi.ci.i.u-tokyo.ac.jp

Yutaka Matsuo
The University of Tokyo
matsuo@biz-model.t.u-tokyo.ac.jp

Mitsuru Ishizuka
The University of Tokyo
ishizuka@i.u-tokyo.ac.jp

Toshio Yokoi
Tokyo University of Technology
yokoi@media.teu.ac.jp

Abstract

Confronting the challenges of annotating naturally occurring text into a semantically structured form to facilitate automatic information extraction, current Semantic Role Labeling (SRL) systems have been specifically examining a semantic predicate-argument structure. Based on the Concept Description Language for Natural Language (CDL.nl) which is intended to describe the concept structure of text using a set of pre-defined semantic relations, we develop a parser to add a new layer of semantic annotation of natural language sentences as an extension of SRL. With the assumption that all relation instances are detected, we present a relation classification approach facing the challenges of CDL.nl relation extraction. Preliminary evaluation on a manual dataset, using Support Vector Machine, shows that CDL.nl relations can be classified with good performance.

1 Introduction

With the dramatic increase in the amount of textual information available in digital archives and on the WWW, interest in techniques for automatically extracting information from text has been growing. Recently, much attention has been devoted to Semantic Role Labeling (SRL) of natural language text with a layer of semantic annotation having a predicate-argument structure. And high-performance systems have been developed using FrameNet[1] and PropBank[5] corpora, respectively, as training and testing materials. As Semantic Role Labeling specifically examines predicate-argument structure, towards the goal of putting the whole sentence into a semantic structural form, Yokoi et al. (2005)[6] presented a descriptive language named Concept Description Language for Natural Language (CDL.nl), which is part of the realization of spirits of the work "semantic information processing"[4]. CDL.nl defines a set of semantic relations to form the semantic structure of natural language sentences in a graphical representation.

In this paper, based on CDL.nl relation set, with the assumption that relation instances have been detected, we describe an algorithm for relation classification with two advantages over previous ones: besides common used lexical information, we use a new lexical resource to provide semantic behavior features of entites; and we use kernel method to model and leverage lexical features separately from syntactic and dependency features to improve the performance. Preliminary experiments are trained on a hand-annotated dataset.

Our contributions can be summarized as follows:

- Our study shows an intermediate phase in the progress to semantic parsing of natural language processing from syntactic processing. Annotation of text with a wider semantic structure can expand the extent to which semantic information can become useful in real Web and NLP applications.
- By modeling and leveraging lexical information separately from syntactic and dependency knowledge, our study also suggests an example of the flexibility of using kernel method to leverage diverse knowledge.

The remainder of this paper is organized as follows. Section 2 introduces the CDL.nl relation set. Section 3 proposes our relation classification method. Section 4 reports the experimental results and our observations. We conclude our work in Section 5.

2 CDL.nl Semantic Relation Extraction Task

Yokoi et al. (2005)[6] presented Concept Description Language for Natural Language (CDL.nl), which is used to describe the semantic/concept structure of text as a core

member of W3C Common Web Language[1]. Two basic elements for describing the structure are Entity and Relation, where the element Entity is used to represent a constituent of sentences with a head word. The entity which heads the relation is called the head entity; the other one is the tail entity. And CDL.nl predefines a set of relation types[2] to represent the semantic structure of sentences. It defines a set of semantic relations containing 44 relation types which are organized into three groups.

Intra-event relation: Relations defining case roles, which are divided into the six abstract relations of *QuasiAgent*, *QuasiObject*, *QuasiInstrument*, *QuasiPlace*, *QuasiState*, and *QuasiTime*. Furthermore, each abstract relation includes several concrete relations which express concrete semantic information. For example, *QuasiAgent* contains five concrete semantic relations: *agt* (agent), *aoj* (thing with attribute), *cag* (co-agent), *cao* (co-thing with attribute), *ptn* (partner).

Qualification relations: In addition to case-specific relation types, CDL.nl also defines qualification relation types. There are nine qualification relations, collectively containing *mod* (modification), *pos* (possessor), and *qua* (quantity). This subset of relations is important to describe an entity with myriad properties.

Inter-entity relations: CDL.nl also defines 13 inter-entity relation types in which both entities assume relatively equal position. Qualification relations collectively contain *con* (condition), *seq* (sequence), *equ* (equivalent), etc. This subset of relations is important to describe a pair of entities with myriad properties.

The CDL.nl relation set is useful to annotate not only facts in sentences about WHO did WHAT to WHOM with WHOM, WHEN, WHERE, WHY, and HOW, but also WHAT has WHICH attributes and restrictions. Fig. 1 illustrates an example showing the graphical structure annotated using CDL.nl relations.

3 Kernel Method for Relation Classification

In this section, facing the challenges of labeling each pair of entities with a specific CDL.nl relation, we describe a relation classification approach which uses kernel functions to model diverse knowledge of three levels of language processing: syntactic analysis, dependency parsing and lexical construction.

3.1 Syntactic Kernel

As a benefit from the Connexor Parser[3], rich linguistic tags can be extracted as features to classify relations between entities. For each pair of entities of relation instances, we extract the following syntactic features and define a syntactic kernel to match two relation instances.

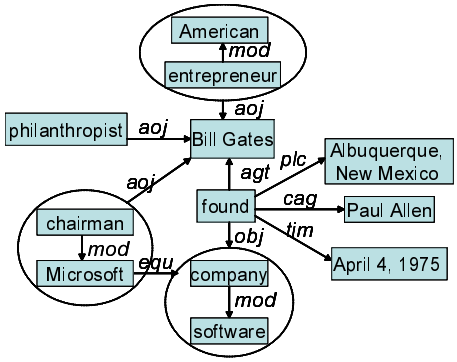

Figure 1. The graphic structure of sentence "*Bill Gates is an American entrepreneur, philanthropist and chairman of Microsoft, the software company he founded with Paul Allen in Albuquerque, New Mexico on April 4, 1975.***"**

Morphology Features: Morphological information tells the details of word forms used in text.

We use a vector to represent the morphology feature space: $X_{Morp} = (x_1, x_2, ..., x_{70})$. Where x_i corresponds to a tag and receives "0" or "1" value, and Connexor Parser defines 70 morphology tags. For each entity E, $X_{Morp}(E) = (x_{e1}, x_{e2}, ..., x_{e70})$. Where, $x_{ei} = 1$ if the set of morphology tags of the headword of E contains the tag of the *ith* position, all other tags not contained will be set to $x_{ei} = 0$.

Syntax Features Syntax describes both surface syntactic and syntactic function information of words. For example, *%NH* (nominal head) and *%>N* (determiner or premodifier of a nominal) are surface syntactic tags, *@SUB* (Subject) and *@F-SUBJ* (Formal subject) are syntactic function tags. The Connexor Parser defines 40 Syntax tags.

As dealing with morphology features, syntax features for an entity E are represented in a vector: $X_{Syn}(E) = (x'_{e1}, x'_{e2}, ..., x'_{e40})$. For two entities of a relation instance R, the syntactic feature vector $X(R)$ is defined as the concatenation of morphology and syntax vector:

$$X(R) = (X_{Morp}(E_1)X_{Morp}(E_2)X_{Syn}(E_1)X_{Syn}(E_2))$$

Then we define a syntactic kernel to match syntactic features between two relation instances R_1, R_2 by simply calculating the dot product of two vectors:

$$K_S(R_1, R_2) = X(R_1) \bullet X(R_2)$$

3.2 Dependency Kernel

For each pair of entities of relation instances, to extract a dependency feature set F_D, we define a dependency token $DT = (dep, path)$, where dep contains two labels: one is

[1]http://www.w3.org/2005/Incubator/cwl/
[2]http://www.miv.t.u-tokyo.ac.jp/mem/yyan/CDLnl/
[3]www.connexor.com

the first depend label in the dependency path, which is governed directly by the headword of head entity; the other is the final label in the dependency path pointing to the headword of participant entity. Both are closest to representing the direct dependency functions of the entity pair. In addition, $path$ is the path in the parse tree from the head entity to the other entity. We define a dependency kernel to match dependency features between two relation instances R_1, R_2 by matching the values:

$$K_D(R_1, R_2) = \sum_{i=1,2} I(DT_{1i}, DT_{2i})$$

Where $I(x, y)$ is a binary string match operator that gives 1 if $x = y$ and 0 otherwise.

3.3 Lexical Kernel

3.3.1 WordNet

WordNet[2] is an on-line lexical system whose smallest unit is "synset", i.e. an equivalence class of word senses under the synonym relation. Synsets are organized by semantic relations such as Synonymy, Antonymy and Hyponymy.

A vector W is defined to capture sense features containing word sense and hypernym senses of the headword of each entity: $W = (w_1, w_2..., w_n)$. Since each word might have many hypernym senses, in our experiments, we select the top four senses.

3.3.2 UNLKB

Based on the CDL.nl semantic relation set, for each usage of the word, we define semantic behavior as a series of CDL.nl semantic relations in which the word participates. Because many words have different senses and usages they might have several semantic behaviors. The UNLKB[4] is a lexicon which organizes words in a hierarchical structure according to their semantic behaviors.

Here are some word-behavior pairs of word *give* in UNLKB:

give(agt>thing,obj>thing)
give(agt>thing,gol>person,obj>thing)
give(agt>thing,gol>thing,obj>thing)
give(agt>volitional thing,obj>action)

The word *give* has semantic behaviors of at least these four kinds. Furthermore, for the second behavior, it has *agent* relation with a thing-type word, *goal* relation with a person-type word and *object* relation with a thing-type word. Here, the type of a word is a hypernym word of the word.

A vector U is defined to capture the hierarchy hypermym of semantic behaviors of word: $U = (u_1, u_2..., u_n)$.

[4]www.undl.org/unlsys/uw/unlkb.htm

3.3.3 Lexical Kernel Development

Both resources - WordNet and UNLKB - encoding different kinds of knowledge and we plan to extract two kinds of features: word senses and semantic behavior of words.

To explicitly capture these features, for the entity pair E_1, E_2, a new lexical feature vector $Y(R)$ is defined as the concatenation of both above lexical vectors:

$$Y(R) = (W(E_1)W(E_2)U(E_1)U(E_2))$$

Then we define the kernel to match lexical features between two relation instances R_1, R_2 by simply calculating the dot product of two vectors:

$$K_L(R_1, R_2) = Y(R_1) \bullet Y(R_2)$$

3.4 Composition Kernel

Having defined all the kernels representing syntactic, dependency and lexical processing results, we develop a composite kernel to combine and leverage individual kernels:

$$K = \alpha K_S + \beta K_D + \gamma K_L$$

4 Experiments

4.1 Experimental Setting

Because this is the first work to extract CDL.nl relations from plain form text, for the training and testing of our method, we use a manual dataset[5] which took 46 person-days of manual annotation and correction effort. 1700 sentences from Wikipedia documents were annotated with 13487 CDL.nl relations including 44 relation types. We evaluated the systems using ten-fold cross validation using this dataset.

To evaluate the performance of our relation classification method, we use one-vs.-all scheme in which each binary classifier will be trained for each relation label. The classifier evaluation is carried out using SVM-light software[3] with our syntactic, dependency, and lexical features.

4.2 Preliminary Experimental Results

The goals of our experiments are twofold: firstly, we intend to study the performance of individual kernels and watch if adding kernels continuously improves the performance. Secondly, we study how to leverage among syntactic, dependency and lexical features to get the best performance.

- **Individual Kernel Evaluation**

Firstly we test three individual kernels and the following two simple combination kernels:

$$K_{S+D} = K_S + K_D$$
$$K_{S+D+L} = K_S + K_D + K_L$$

[5]http://www.miv.t.u-tokyo.ac.jp/mem/yyan/CDLnl/

Table 1. Preliminary performance of individual kernels

Kernel	Precision	Recall	F-value
K_S	79.33	85.78	82.43
K_D	**83.62**	**83.56**	**83.59**
K_L	73.49	81.63	77.35
K_{S+D}	**85.63**	**85.91**	**85.77**
K_{S+D+L}	**86.35**	**87.43**	**86.89**

Table 2. Performance of using different weights.

α, β, γ	Precision	Recall	F-value
0.4,0.5,0.6	86.65	87.57	87.11
0.4,0.6,0.5	86.73	88.16	87.44
0.6,0.5,0.4	85.96	87.10	86.53
0.5,0.6,0.4	86.59	88.03	87.31
0.5,0.6,0.5	86.64	87.90	87.26
0.5,0.5,0.5	**86.35**	**87.43**	**86.89**
0.4,0.6,0.4	**86.83**	**88.49**	**87.65**
0.4,0.5,0.4	86.67	88.16	87.41
0.4,0.5,0.3	86.65	88.03	87.34
0.3,0.5,0.4	86.64	88.36	87.49

From Table 1 we can get two observations, one is that using different feature set, the performance is different. The performance of using dependency kernel is the best than using syntax kernel and lexical kernel. Another observation is that adding kernels continuously can improve the performance, which indicates they provide additional clues to the previous setup.

- **Composition Kernel Evaluation**

Then for the composition kernel, we experiment several sets of α, β, γ values to compare the performance. As shown in Table 2, limited times of evaluation on the development set shows that our composition kernel yields better performance when α, β, γ are set to 0.4, 0.6 and 0.4. And it also shows that after fine-tuning parameters, the performance is better than that of using equal weights for all kernels.

Although we do not enumerate all the values of each parameter, we can see that lexical features do not contribute much as expectation to the performance. The reason might be: the WordNet hierarchy is not a tree but rather includes multiple inheritances and a further complication is that several WordNet word-sense pairs or UNLKB word-behavior pairs are possible for a given head word. A word sense disambiguation module capable of distinguishing word senses and word behaviors might improve our results.

In order to compare the contribution of each lexicon, we also evaluate each kind of lexical features. As shown in Table 3, the performance of using semantic behavior features from UNLKB lexicon is improved.

Table 3. Evaluation on incremental lexical features.

	Lexicon	Precision	Recall	F-value
A	No-Lexicon	85.63	85.91	**85.77**
B	A+WordNet	85.80	86.88	**86.34**
C	B+UNLKB	86.35	87.43	**86.89**

Through the preliminary experiments, we can see that despite confronting so many obstacles, CDL.nl relations were classified using our approach with Precision, Recall, and F-values that are, respectively, 86.83, 88.49, 87.65.

5 Conclusions

In this paper, to surmount the challenges of semantic annotation of Web text, we created a new parser that (1) used a new set of semantic relations of CDL.nl, which has better coverage than those of SRL, to represent the semantic structure of text. In addition, (2) we proposed a relation classification approach with two advantages over previous ones first by using a new lexicon to provide semantic behavior features of words, then using kernel method to model lexical features separately from syntactic and dependency features into a composition kernel. Experiments conducted using our manual dataset revealed that CDL.nl relations can be classified with good performance and two advantages of our method actually improved the performance of relation classification.

The immediately extension of our work is to improve the performance of relation classification by enriching the dataset, we plan to bootstrap the classifiers to get larger amount of data.

References

[1] C. F. Baker, C. J. Fillmore, and J. B. Lowe. The Berkeley FrameNet Project. *In Proc. COLING-ACL-98.*

[2] C. Fellbaum. WordNet: An electronic lexical database. *Cambridge, MA: MIT Press, 1998.*

[3] T. Joachims. Text Categorization with Support Vector Machine: learning with many relevant features. *In Proc. ECML-98.*

[4] M. Minsky. Semantic Information Processing. *MIT Press, Cambridge, MA.*

[5] M. Palmer, D. Gildea, and P. Kingsbury. The proposition bank: An annotated corpus of semantic roles. *Computational Linguistics, vol. 31, no. 1.*

[6] T. Yokoi, H. Yasuhara, H. Uchida, et al. CDL (Concept Description Language): A Common Language for Semantic Computing. *In WWW2005 Workshop on the Semantic Computing Initiative (SeC2005).*