Open Information Extraction for SOV Language based on Entity-Predicate Pair Detection

$Woong-Ki\ Lee^1$ $Yeon-Su\ Lee^1$ $Hyoung-Gyu\ Lee^1$
$Won-Ho\ Ryu^2$ $Hae-Chang\ Rim^1$
(1) Department of Computer and Radio Communications Engineering
Korea University, Seoul, Korea
(2) Software R&D Center, Samsung Electronics Co., Ltd.
Suwon-si, Gyeonggi-do, Korea
{wklee,yslee,hglee,rim}@nlp.korea.ac.kr
wonho.ryu@samsung.com

ABSTRACT
Open IE usually has been studied for English which of one of subject-verb-object(SVO) languages where a relation between two entities tends to occur in order of entity-relational phrase-entity within a sentence. However, in SOV languages, two entities occur before the relational phrase so that the subject and the relation have a long distance. The conventional methods for Open IE mostly dealing with SVO languages have difficulties of extracting relations from SOV style sentences.
In this paper, we propose a new method of extracting relations from SOV languages. Our approach tries to solve long distance problems by identifying an entity-predicate pair and recognizing a relation within a predicate. Furthermore, we propose a post-processing approach using a language model, so that the system can detect more fluent and precise relations. Experimental results on Korean corpus show that the proposed approach is effective in improving the performance of relation extraction.

KEYWORDS: relation extraction, SOV language, predicate extraction.

1 Introduction

Relation extraction detects the relationship between entities from natural language text and makes the information as a structured data. In the traditional relation extraction task, the relationship that needs to be extracted is pre-defined, according to the domain specific goal. Recently, the World Wide Web provides vast amounts of documents and internally accumulates a variety of valuable relational information. Therefore, extracting and utilizing the information from a large web corpus become a hot research issue.
Banko et al. (2007) announced the first proposed system as a new paradigm called Open IE. The goal of Open IE system is to extract all possible correct relationships between entities without pre-defining the relationships. Open IE has shown successful result in some degree. Hereafter, a lot of research has been carried out on Open IE like TextRunner (Yates et al., 2007), REVERB (Etzioni et al., 2011) and WOE (Wu and Weld, 2010). However, most previous approaches have been proposed for English corpus. Although they are language independent approaches, when applied to another kind of language such as Korean, an unexpected problem occurs. English is a SVO(Subject-Verb-Object) word order language. In most cases, a relational phrase appears between subject(an entity) and object(another entity). Therefore, they naturally assume that the phrase is associated with the subject entity.
However, in SOV language such as Korean, Japanese and Turkish, by default, the subject, object, and verb usually appear in that order. And the modifiers should always be placed before their modificands. Moreover, the word order is relatively free. Therefore, it is difficult to extract a relationship by using English like assumption. The following example is a sentence from a newspaper article.

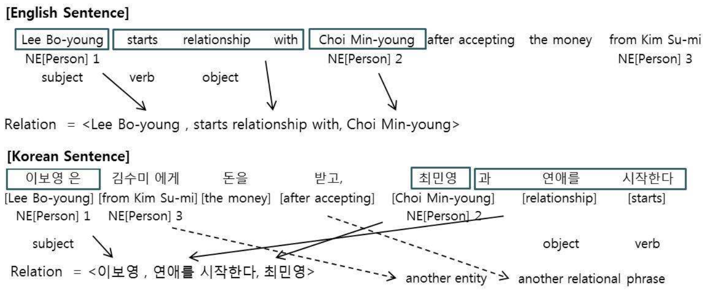

Figure 1: Word ordering and relation extraction: SVO vs. SOV

In Figure 1, the relational phrase of "이보영(Lee Bo-young[PERSON])" is "돈을 받고(accepting the money from)" and "연애를 시작한다(starts relationship with)". The noticeable difference from the English sentence is that the entity, "이보영(Lee Bo-young[PERSON])", and the relational phrase, "연애를 시작한다(starts relationship with)", is far apart. Moreover, before appearing the phrase, another irrelevant entity, "김수미(Kim Su-mi[PERSON])", and the relational phrase, "돈을 받고(accepting money from)", appear. Because of these characteristics, it is impossible to apply the assumption of English (a relational phrase appears between two entities) to Korean sentences. In the strict sense, there exist two relational tuples in this sentence. Another is <Lee Bo-young, accepting the money from,

Kim Su-mi>. However, it is also impossible to extract this tuple by the assumption in English. To solve the problem, we consider the connectivity between entities and predicate in the first place. And then, we identify if there exist a relationship in the connected tuple. In addition, to extract a comprehensible phrase, we apply a language model to the relation tuple.

2 Open IE System for SOV Language

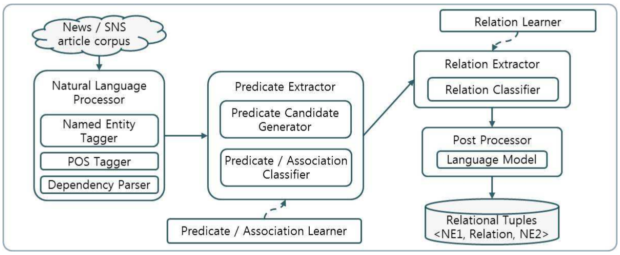
News / SNS article corpus
Natural Language Processor
Named Entity Tagger
POS Tagger
Dependency Parser
Predicate Extractor
Predicate Candidate Generator
Predicate / Association Classifier
Predicate / Association Learner
Relation Learner
Relation Extractor
Relation Classifier
Post Processor
Language Model
Relational Tuples <NE1, Relation, NE2>

Figure 2: System architecture

Figure 2 shows the architecture of our proposed system. It takes a web corpus as an input and a set of relation tuples <NE1, Relation, NE2> as an output. First, the corpus is pre-processed. The named entities in a sentence are recognized and the sentence is POS-tagged and parsed to a dependency tree.

The relation extraction process comprises two key modules. The first module, *predicate extractor* generates <entity, predicate> pair candidates by using a simple POS constraint and classifies whether the predicate is a proper one of the entity by using the maximum entropy classifier. The second module, *relation extractor* finds another entity in an entity-predicate pairs that are classified into a correct one at the previous step. And then, it transforms two entities and the predicate into a candidate relation tuple form. Finally, the maximum entropy classifier decides if the candidate relation tuple represents a correct relationship between two entities.

2.1 Predicate Extraction

2.1.1 Predicate Candidates Generation

First, we generate predicate candidates that can be connected for one NE. We assume that the "predicate" is based on a verb phrase. We follow the Etzioni et al. (2011)' verb-based constraints to prevent the relational phrase from being incomprehensible. The predicate candidates are generated through the proposed algorithm, as shown in Figure 3. The start point of a predicate is a NP chunk and the end point is a verb or the end of a sentence. The reasons of extracting both the verb and the adjacent NP chunk are as follows. 1) In SOV language, the subject is far away from the verb. However, the other entity (an object or an adverb) is adjacent to the verb. 2) We restrict the predicate to the one including a relational meaning.

2.1.2 Entity-Predicate Pair Detection

The predicate classifier is the part of deciding whether the predicate in an entity-predicate pair extracted from the predicate generator describes about the entity or not. In Figure 1,

```
Define Set: StartSet, EndSet, CandidateSet
for( i=0;  i< sentence.length;  i++ )
        if( (POS(i),...)  match "(adjective | noun)* (noun)" )
                put i to StartSet
for( j=0;  j< sentence.length;  j++ )
        if( POS(j)  match "(verb | ending)" )
                put j to EndSet

for( i=0;  i  <  StartSet.size;  i++ )
        for( j=0;  j  <  EndSet.size;  j++ )
                if( StartSet[i]  <  EndSet[j] )
                        put < i, j > to CandidateSet
```

Figure 3: Predicate candidate generation algorithm

according to the predicate generation pattern, candidates are generated as follows. P1:<이보영(Lee Bo-young[PERSON]), 최민영과 연애를 시작한다(starts relationship with Choi Min-young[PERSON])>, P2:<김수미(Kim Su-mi[PERSON]), 최민영과 연애를 시작한다(starts relationship with Choi Min-young[PERSON])>, etc. In this case, the predicate of P1 is a correct description about the entity, Lee Bo-young. But the predicate of P2 is an inappropriate description about the entity, Kim Su-mi. The goal of the predicate classifier process is to detect a correct or incorrect connection between an entity and a predicate, to improve the performance of relation extraction.
For classifying correct entity-predicate pairs, we use Maximum Entropy classifier which is one of Machine Learning method. The classifier assigns probability that the predicate is correct and is connected to the entity. Classifier is learned by supervised learning and uses the following features in Table 1. The features are grouped into three major categories.

Surface	Syntactic	Semantic
· Sentence length · Predicate length · Distance NE1 ∼ predicate · # of other entities in sentence · Existence of verb between NE1 and predicate · Position of NE1 and phrase	· Functional words next to NE1 · POS tags in predicate · POS tags in left/right side of predicate · POS tags at the boundary of predicate · The length of dependency link between NE1 and predicate	· NE1 type · Verb type · Existance of matched verb frame

Table 1: Surface, syntactic and semantic features used

In order to investigate the connection between an entity and a predicate, we use the distance between an entity and a predicate, whether there is a verb between an entity and a predicate, the postposition next to an entity etc. as a feature. And for the purpose of identifying the suitability of a predicate as description, we use the predicate length, POS tag of predicate and others. The dependency link is the number of links between the entity and predicate in dependency tree. The type of entity indicates the type that the entity belongs to, like "PERSON", "PROGRAM", "LOCATION" and so forth. We build a set of verb-arguments frames for each high-frequent verb by using the collocation of a verb and argument types. We decide the argument type by using the functional words attached. There are four argument types: S(Subject), O(Object), B(Adverbial) and C(Complement). We assign same type to the verbs which have the same frame. The feature presents both whether the predicate includes some important information and whether the argument and the predicate are connected semantically.

2.2 Relation Extraction

In this step, we extract relation tuples from the entity-predicate pairs. First, we separate the other entity, NE2, from the predicate and convert the NE1, NE2, and the rest phrases of predicate to a relation tuple form. And we classify whether the tuple is a correct relation tuple or not. For example, from an entity-predicate pair, <신세경(Sin Se-kyoung[PERSON]), 인기 프로그램 무한도전에 출연한다(make an appearance on the famous TV program Muhan-dojeon[TV PROGRAM])>, we can get two relation tuples, R1:<신세경(Sin Se-kyoung[PERSON]), 인기 프로그램(famous TV program), 무한도전(Muhan-dojeon[TV PROGRAM])>와 R2:<신세경(Sin Se-kyoung[PERSON]), 출연한다(make an appearance on), 무한도전(Muhan-dojeon[TV PROGRAM])>. Among the R1 and R2, only the R2 is a correct tuple. To determine that the candiate relation tuple is correct, we use the Maximum Entropy classifier learned by supervised learning. Like the predicate classifier, we use several surface, syntactic and semantic features. Additionally, to judge that the triple has a relationship, we uses the following features in Table 2.

Surface	Syntactic	Semantic
· The precedency between the NE2 and the rest phrase · Position of NE2	· Functional words next to NE2 · POS tags in left/right side of two entities	· Type of NE2

Table 2: Additionally used features for relation extraction

2.3 Post-processing Using Language Model

We propose a new post-processing method using a language model, in addition to the relation extraction method based on the predicate extraction. The LM-based approach is motivated by the following common errors, which may be incorrect relations in spite of high probabilities given by our relation classifier. For example, the tuple <박명수 (Park Myoung-su[PERSON]), 평화가 찾아왔다 (the peace comes to), 무한도전(Muhan-dojeon[TV PROGRAM])> can be outputted by the system described in the previous section. These erroneous tuples cannot usually produce a fluent and comprehensible sentence when concatenating their entities and the relational phrase. Therefore, this problem can be solved by using a language model.
We measure the perplexity of the word sequence generated by concatenating two entities and the relational phrase in order of the occurrence in its original sentence. The relation tuples that have a higher perplexity than a threshold are removed from the final set of relation tuples. To construct the Korean 5-gram language model, we use the refined Sejong corpus (Kang and Kim, 2004) consisted of 6,334,826 sentences.

3 Experiments

3.1 Experimental Environment

We have experimented with our method on the Korean news corpus crawled in television program domain from August 13, 2011 to November 17, 2011. The corpus consists of 118K articles and 11.4M sentences. We have performed named entity recognition for pre-processing of this news corpus and have sampled 7,686 sentences containing one or more entities from the NE-recognized corpus. Among these annotated sentences, 4,893 sentences, 2,238 sentences, and 555 sentences are used as the training set for the entity-predicate pair detection, the training set for relation extraction, and the test set for relation extraction, respectively. We used the precision, the recall, and the f-measure as evaluation metrics. When matching between a gold relational phrase and a system output to measure these metrics, we adopted the relaxed matching in which the difference between two target phrases boundaries was permitted up to two words on both the left and the right of each phrase.

3.2 Evaluation of Relation Extraction

We first evaluated the effectiveness of the entity-predicate pair detection. We compared the performance of extracting the relation tuples after the proposed detection phase with the performance of the baselines that extract the tuples without the phase. We set two baseline systems. The first is the system that passes all possible entity-predicate pairs to the relation extraction phase after the predicate candidate generation (ALL). The second is the system that passes only the entity-predicate pairs including the nearest entity for each predicate candidate to the relation extraction phase (NEAR). Table 3 shows the performance of relation extraction of the baseline systems and the proposed system. In this experiment, the classification threshold was optimized on the development set by using the f-measure as the objective function. As shown in Table 3, the proposed approach outperformed both the baseline systems. These results show that our additional phase is effective in finding the relation between two entities in SOV language such as Korean. This also shows that it is helpful to consider first the relationship between an entity and its predicate, prior to recognizing the relationship between two entities.

System	Precision	Recall	F-measure
ALL	0.4540	0.4726	0.4631
NEAR	0.5177	0.5000	0.5087
Proposed	0.5223	0.5616	0.5413

Table 3: Effectiveness of the entity-predicate pair detection phase

4 Demo

Our Open IE system's two key modules, the predicate extractor and the relation extractor were implemented in C++. In addition, to provide users with searching service, we developed extra modules, a ranker and a viewer. The predicate extractor and the relation extractor work periodically as a batch job. However the ranker and the viewer work on demand. Optionally, users can enter the duration, an entity name or a (part of) relation. Then the system looks up the relation DB and shows the following results.
· Relation tuple view
· Relation tuple view including a user-queried entity
· Entity view related to a user-queried relational phrase
· Entity view related to a user-queried entity
We implemented the searching service as a web application.

5 Conclusions

In this paper, we propose a new Open IE system for SOV language. In order to extract relation in SOV language, the system extracts entity-predicate pairs by considering the connectivity between an entity and a predicate, prior to identifying a relationship between two entities. We have shown that our system is effective in improving the performance of relation extraction. The paper's contributions are as follows. First, though the word order is relatively free or there is long distance between an entity and a predicate, the relation is extracted successfully. Second, our post-processing approach using a language model has an effect on finding more fluent and precise relations.

Acknowledgments

This work was supported by Software R&D Center, SAMSUNG ELECTRONICS Co., Ltd.

References

Banko, M., Cafarella, M., Soderland, S.and Broadhead, M., and Etzioni, O. (2007). Open information extraction from the web.

Etzioni, O., Fader, A., Christensen, J., Soderland, S., and Center, M. (2011). Open information extraction: The second generation. In *Twenty-Second International Joint Conference on Artificial Intelligence*.

Kang, B.-M. and Kim, H. (2004). Sejong korean corpora in the making. In *Proceedings of LREC 2004*, pages 1747–1750.

Wu, F. and Weld, D. (2010). Open information extraction using wikipedia. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 118–127. Association for Computational Linguistics.

Yates, A., Cafarella, M., Banko, M., Etzioni, O., Broadhead, M., and Soderland, S. (2007). Textrunner: open information extraction on the web. In *Proceedings of Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, pages 25–26. Association for Computational Linguistics.