Letter

Extracting Events from Web Documents for Social Media Monitoring using Structured SVM

Yoonjae Choi[†], ***Nonmember***, **Pum-Mo Ryu**[†], ***Nonmember***, **Hyunki Kim**[†], ***Nonmember***, **Changki Lee**[††,*], ***Member***

SUMMARY Event extraction is vital to social media monitoring and social event prediction. In this paper, we propose a method for social event extraction from web documents by identifying binary relations between named entities. There have been many studies on relation extraction, but their aims were mostly academic. For practical application, we try to identify 130 relation types that comprise 31 predefined event types, which address business and public issues. We use structured Support Vector Machine, the state of the art classifier to capture relations. We apply our method on news, blogs and tweets collected from the Internet and discuss the results.
key words: *Relation Extraction, Structured SVM, Natural Language Processing, Information Extraction*

1. Introduction

Relation extraction is an important part of information extraction from natural language text. One typical example is using relations between entities in a question answering system [14]. Relation extraction could also be applied to social media monitoring systems [1,2]. The recent explosion of user participation in Web 2.0 has produced a huge amount of unstructured text information on the Internet. Extracting useful relations from such information could lead to detection of meaningful or noteworthy social events, such as release of a new product or rise/decline of stock prices.

Researchers have studied relation extraction for some time. But most studies focused on developing algorithms to increase the performance on widely used corpora with several predefined relations or to extract unspecified relations in an unsupervised manner. This paper, on the other hand, tries to capture relations of 130 types in order to recognize 31 different types of social events.

The rest of this paper is organized as follows: Section 2 discusses related work and contribution of this paper. Section 3 presents our method that involves structural support vector machine. Settings and results of our experiments are given in section 4. And finally, we finish this paper with conclusion and future work in section 5.

Manuscript received January xx, 20xx.
Manuscript revised March xx, 20xx.
†The authors are with ETRI, Daejeon, 305-700, South Korea.
††The author is with Kangwon National University, Chuncheon, Kangwon-do, 200-701, South Korea.
*Corresponding author. Tel.: +82-33-250-8443.
E-mail: leeck10@gmail.com

2. Related Work

Previous work on relation extraction can be largely divided into supervised learning and unsupervised learning. Roth and Yih [3] belongs to the former. They tried to classify relations between entities using loopy belief propagation. While their approach was sophisticated, their work was mostly experimental with only 3 relations: kill, born_in, other_rel.

Kambhatla [4] uses maximum entropy classifier, which is also a supervised model. His work, however, was applied to ACE 2003 RDC data, which uses 24 relations types which are too general and ambiguous for our task. While maximum entropy is a very popular classifier, we employ structured SVM, which shows better performance in various academic fields [16].

Zelenko et al. [5], Culotta and Sorensen [6], Zhao and Grishman [7], Zhou and Zhu [8] used various kinds of kernels. Since using kernel methods in structured SVM decreases inference speed, we chose to employ linear kernels only, considering the practical environment in which we will be processing millions of documents per day.

The work of Ravichandran and Hovy [9], Pantel and Pennacchiotti [10] fall into minimally supervised method category. They used bootstrapping methods to automatically learn relation patterns. But it is difficult to deal with long distance dependencies when using surface patterns only [14]. And the more patterns you use, the slower the system becomes.

Banko et al. [11], Banko and Etzioni [12], introduced unsupervised relation extraction frameworks. They constructed training corpus automatically and trained a classifier to recognize relations between entities. While their approach requires very little human effort, the relations harvested cannot be used directly, because relations were not defined from the beginning. Therefore in our task, where one has to capture specific relations, unsupervised approach is not an appropriate method to adopt. Wang et al. [13] also employs an unsupervised method. They borrow relation types from DBPedia and automatically construct a training corpus. Their approach also forbids us to define our own specific relation types.

This work is an expansion of Lee et al.'s work [14].

Their work, which employs supervised method, used 37 types of relations, such as has_position, has_person, and has_email. While their relation types are suitable for general relation extraction on PLO(Person, Location, Organization) field, our goal is to detect social events, which drove us to define various types of specific relations.

This work tries to achieve social event extraction using relation extraction methods and make the following contributions.

- With the social media monitoring system in consideration, we put relation extraction to a very practical situation. Unlike other works, which use a small number of rather vague relation types, we use 130 types that capture very specific, event-related information.
- We use a novel feature generated by a multiclass classifier called "Event Detector". Since our last work [14] was not targeted for social event extraction, we developed Event Detector to adapt the system to the new task. This method showed around 2% F-score improvement of our system.
- We test various kinds of SVM models using documents of different media, namely news, blog, and tweet. Such experiment gives insight into the relationship between training methods and the characteristic of documents. It also suggests which model to use for each document type.

3. Relation Extraction

We defined 31 types of social events to address business and public issues. The ten most frequently appearing event types[1] and their examples are given in Table 1. Each event consists of binary relations. For example, "Product Release" has "product release organization", "product release date", "product release price" as its member. Some events, such as "Vote" has only one binary relation, which is "vote_date". For total 31 events, there are 130 binary relations. By classifying the relations between two entities into one of 130 binary relation types, we can build up events, which is a similar process to template slot filling.

We assume that entities in sentences are tagged and given to us[2]. Our entity types consist of 15 categories: person, study field, theory, artifacts, organization, event, location, civilization, date, time, quantity, animal, plant, material, and terms. Each category is further divided into subcategories, giving us in total 180 classes. For example, organization type has "org_religion", "org_education", "org_politics", etc. We then apply POS(Part-of-speech) tagger and dependency parser that we have developed. Using the output of the pipeline, we construct features to use in classification.

Table 1 Ten most frequent events

Event	Example
Announce	President Lee Myungbak said that …
Event Open	Apple held WWDC 2012 in San Francisco
Stock Price Decline	Samsung plummeted 10% from …
Stock Price Rise	Samsung's stock rose 1.5% from…
Company Legal Act	Samsung was ordered to suspend business…
Employment Change	Apple employed Tim Cook as the CEO…
Policy Enact	The gov. installed "Free School Meal" law in…
Credit Rate	South Korea was rated AA+ by S&P…
Product Release	Samsung released Galaxy S3 this summer…
Results Announcement	Samsung announced $130 million net profit…

[1] The frequencies were counted from event test sets in section 4.

The features we used are commonly used in relation extraction task: morphemes around two entities, part-of-speech tags, distance between two entities, number of words between two entities, entity types of two entities, and parse tree information. Section 3.2 briefly discusses how parse tree information was used as a feature. We also use the output of Event Detector, which will be discussed in section 3.3, to provide global information of the input sentence.

3.1 Structural SVM

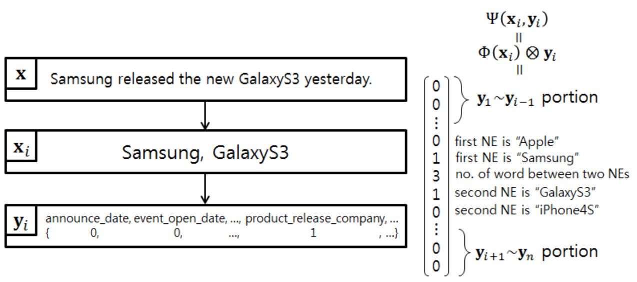
Our task is essentially a classification problem: to identify a pair of entities as one of 131 classes, 130 of which are relation types and 1 being no-relation. To perform this task we use structured SVM [15], the state-of-the-art machine learning technique. Unlike plain SVM, which is only capable of binary classification, structured SVM can learn from structured, interdependent output space. Some use cases of structured SVM are sequence labeling, sentence parsing, and hierarchical classification.

Let $\mathbf{x}_i$ be the i-th entity pair in the sentence. Let $\mathbf{y}_i$ be the answer relation for $\mathbf{x}_i$. Structural SVM tries to solve the following quadratic program:

$$\min_{w,\xi} \frac{1}{2} \parallel \mathbf{w} \parallel^2 + \frac{C}{n}\sum_i^n \xi_i\,, \quad s.t.\ \forall i,\ \xi_i \geq 0 \qquad (1)$$
$$\forall i, \forall \mathbf{y} \in \mathrm{Y} \backslash \mathbf{y}_i : \mathbf{w}^{\mathrm{T}} \delta\Psi_i(\mathbf{x}_i, \mathbf{y}) \geq L(\mathbf{y}_i, \mathbf{y}) - \xi_i$$

where $\delta\Psi_i(\mathbf{x}_i,\mathbf{y}) = \Psi_i(\mathbf{x}_i,\mathbf{y}_i) - \Psi_i(\mathbf{x}_i,\mathbf{y})$, $\Psi_i(\mathbf{x}_i,\mathbf{y}_i)$ is a joint feature function, $(\mathbf{x}_i,\mathbf{y}_i)$ is a training example and $L(\mathbf{y}_i,\mathbf{y})$ is a 0-1 loss function. An example of $\Psi_i(\mathbf{x}_i,\mathbf{y}_i)$ is show in Figure 1. In the case of multiclass classification, $\Psi_i(\mathbf{x}_i,\mathbf{y}_i)$ is defined as the tensor product of $\Phi(\mathbf{x}_i)$ and $\mathbf{y}_i$ as shown in Figure 1, where $\Phi(\mathbf{x}_i)$ is a feature representation of a training instance. The weight $\mathbf{w}$ in Equation (1) is the gradient of the hyperplane that maximizes the margin between the output classes. $\mathbf{w}$ can be trained by using cutting-plane algorithm [15] or its variants [16,17,18].

[2] For this paper, named entities in the data set were manually annotated.



$\Psi(\mathbf{x}_i, \mathbf{y}_i)$
$\Phi(\mathbf{x}_i) \otimes \mathbf{y}_i$
x
Samsung released the new GalaxyS3 yesterday.
$\mathbf{x}_i$
Samsung, GalaxyS3
$\mathbf{y}_i$
announce_date, event_open_date, ..., product_release_company, ...
{ 0, 0, ..., 1, ...}
$\mathbf{y}_1 \sim \mathbf{y}_{i-1}$ portion
first NE is "Apple"
first NE is "Samsung"
no. of word between two NEs
second NE is "GalaxyS3"
second NE is "iPhone4S"
$\mathbf{y}_{i+1} \sim \mathbf{y}_n$ portion

Figure 1 Example of $\Psi(\mathbf{x}, \mathbf{y})$ for Relation Extraction

3.2 Parse Tree Information

The events we aim to extract are strongly hinted by a specific verb in a given sentence. For example, "Product Release" event would use verbs such as released, sell, or developed. Therefore, given a parse tree of a sentence, we use the lowest common ancestor(LCA) of two entities to help the classifier decide whether they should form a specific relation.

Given a sentence "Samsung released Galaxy S3 on July 30th", the LCA of "Samsung" and "Galaxy S3" is "released". Therefore we can think there is a good chance that "Samsung", which is an organization, and "Galaxy S3", which is a product, will form a "product_release_organization" relation. The same rule applies for "Galaxy S3" and "July 30th", to form a "product_release_date" relation.

3.3 Event Detector

Since our task involves 31 types of social events, it would be helpful to give a hint to the classifier to which event a given sentence belongs. Therefore, we developed Event Detector which classifies a given sentence into one of 32 classes, 31 of which are event types and one additional no-event. One can consider such information as providing global dependency for a given entity pair.

We used structured SVM for classification. The features we used are bigrams and common syntactic information such as length of the sentence, number of alphabets and symbols, number of digits, number of unique terms. Implementation and training are done in a similar way depicted in section 3.1. Details are omitted due to limited space.

Event Detector takes sentences as input, so we use it before relation extraction. The result of Event Detector is stored in a global variable and used as a feature each time we perform relation extraction on a pair of entities in the same sentence.

4. Experiment and Results

We used Korean documents collected from the Internet to train structured SVM. The documents consist of 15295 sentences from news article, 6776 sentences from blog articles, 5713 tweets from Twitter. We take 1376 news sentences, 745 blog sentences, 543 tweet sentences to use as test sets. We divided the corpus to 9:1 ratio by file size. The documents were all manually labeled. Sentences to train Event Detector were also extracted from the same documents.

The news training set has total 379115 entity pairs. Among them are 26250 instances of 130 relation types, "announce_date" being the most frequent with 2116 instances and "product_release_price" being the least frequent with 7 instances. The news test set has 30635 entity pairs. Among them are 2144 instances of 130 relation types, "announce_date" being the most frequent with 171 instances and "lawsuit_date" being the least frequent with 1 instance. Among 417710 entity pairs, the blog training set has 3801 instances of 116 relation types, "announce_date" being the most frequent with 458, "investigate_date" being the least frequent with 1. Among 29667 entity pairs, the blog test set has 435 instances of 66 relation types, "stock_rise_point" being the most frequent with 66, "invest_price" being the least frequent with 1. Among 100358 entity pairs, the tweet training set consists of 2509 instances of 110 relation types, "policy_opposition_person" being the most frequent with 585, several relations such as "price_rise_company" and "product_release_price" appearing only once. Among 11197 entity pairs, the tweet test set has 346 instances of 70 relation types, "policy_opposition_person" being the most frequent with 73, several relations such as "award_date", "stock_rise_date", and "invest_target" appearing only once.

We trained various types of models to find the best model for each media: news, blog, and tweet. We used stochastic subgradient descent method [18] for training. We first trained a model using the entire training corpus, which we will call "Hybrid Model". We also trained three separate models using news, blog, and tweet training set. We will call them "News Model", "Blog Model", and "Tweet Model", respectively

Table 2 shows the accuracy of Event Detector. We evaluated Hybrid Model on each media's test set. News

Table 2 Performance of Event Detector

Model	News Test Set	Blog Test Set	Tweet Test Set
Hybrid	93.74	99.11	97.21
News	94.4	-	-
Blog	-	85.62	-
Tweet	-	-	87.78

Table 3 Comparison of performance before/after using Event Detector (Assuming entities are given)

Model	News Test Set		Blog Test Set		Tweet Test Set	
	Before	After	Before	After	Before	After
Hybrid	69.1	71.2	63.9	65.0	69.33	71.9
News	68.73	71.5	-	-	-	-
Blog	-	-	62.8	62.9	-	-
Tweet	-	-	-	-	67.2	68.7

Table 4 Performance of relation extraction (Assuming entities are given)

Model	News Test Set			Blog Test Set			Tweet Test Set		
	P	R	F	P	R	F	P	R	F
Hybrid	75.3	67.5	71.2	71.2	59.7	65.0	78.1	66.6	71.9
News	74.3-	68.9	71.5	67.2	45.7	54.4	67.7	50.9	58.1
Blog	-	-	-	72.6	55.6	62.9	-	-	-
B.Adapt	-	-	-	71.3	60.7	65.6	-	-	-
Tweet	-	-	-	-	-	-	75.8	62.8	68.7
T.Adapt	-	-	-	-	-	-	75.8	64.1	69.5

Model, Blog Model, and Tweet Model are also evaluated on their respective test set. Hybrid Model performs incredibly well on blog and tweet test set. News Model slightly outperforms Hybrid Model on news test set. Therefore, we used News Model when performing relation extraction on news data, and Hybrid Model for blog and tweet data.

Table 3 shows the F-score of relation extraction before and after using Event Detector. As before, we evaluated Hybrid Model on each media's test set, News Model, Blog Model, and Tweet Model on their respective test set. The impact of Event Detector on the performance is evident according to the result. We therefore assume the use of Event Detector for the rest of this paper.

Two additional models are trained using a domain adaptation method, specifically, a prior model [20]. This method uses the model trained only on the source data as a prior on the weights for a new model trained on the target data. Therefore the new model "prefers" to have weights that are similar to the weights from the basis model, unless the data demands otherwise. Domain adaptation using the prior model tries to solve the following quadratic program:

$$\min_{w,\xi} \frac{1}{2} \parallel \mathbf{w} - \mathbf{w}_0 \parallel^2 + \frac{C}{n} \sum_i^n \xi_i \,, s.t.\ \forall i,\ \xi_i \geq 0 \qquad (2)$$
$$\forall i, \forall \mathbf{y} \in \mathrm{Y} \backslash \mathbf{y}_i \colon \mathbf{w}^{\mathrm{T}} \delta \Psi_i(\mathbf{x}_i, \mathbf{y}) \geq L(\mathbf{y}_i, \mathbf{y}) - \xi_i$$

where the same notations are used as in (1) except $\mathbf{w}_0$, which is the model trained only on the source data. Since news sentences are likely to be most grammatical, we used news model as the basis model and trained a new model using blog training set. We will call this model "Blog.Adapt Model". The training was done using fixed-threshold sequential minimal optimization method [16]. We also trained a model for tweets using the news model as the basis. We call this model "Tweet.Adapt Model". Blog.Adapt Model is evaluated on blog test set, and Tweet.Adapt Model on tweet test set.

We additionally evaluate News Model on blog test set and tweet test set. This experiment was conducted under the assumption that, news sentences being most grammatical, News Model could be used on other media and exhibit high precision.

Table 4 shows the performance of each model on three types of test set. Hybrid Model outperformed other models in the case of tweet test set. It also showed similar performance to News Model when evaluated on news test set with slightly higher precision and slightly lower recall. Hybrid Model is also comparable to Blog.Adapt Model, which is the best performing model when tested on blog test set. News Model excels only in news test set, proving that our assumption was incorrect. Blog Model is competent in blog test set, showing higher precision and lower recall when compared to Hybrid Model and Blog.Adapt Model. Tweet Model shows lower performance in every aspect when compared to Hybrid Model. Tweet.Adapt Model shows better performance than Tweet Model, but Hybrid Model performs significantly well in tweet test set, making Tweet.Adapt Model obsolete. In overall, Hybrid Model seems to be the most competent, if not the best, in all three situations.

Table 5 Performance of event extraction (Aggregate performance)

Model	News Event Set			Blog Event Set			Tweet Event Set		
	P	R	F	P	R	F	P	R	F
Hybrid	79.9	55.6	65.6	73.3	30.8	43.4	65.3	22.5	33.4

We now look at the performances of several related works from the past. It is important to note that the dataset we have used and the datasets others have used are very different, and we present their performances for reference only. Therefore direct comparison between our method and others would be inappropriate. Kahmbahtla [4] uses ACE 2003 RDC evaluation set, which has 23 relation types, and achieves F-score of 55.2. Culotta and Sorensen [6] also uses the same evaluation set and achieves F-score of 45.8. Zhao and Grishman [7] uses ACE 2004 RDC evaluation set, which consists of 23 relation types, and achieves F-score of 70.35. Zhou and Zhu [8] also uses the same evaluation set and achieves F-score of 77.6. Lee et al.[14] uses Korean corpus, which consists of 37 relation types, and shows F-score of 72.8. Considering that we only use linear kernels, and that there are 130 relation types in our task, and the fact that we also deal with user generated social contents, it would not be absurd to say that our method seems quite competent. The ambiguity between relation types, such as the similarity of "Stock Rise" event and "Price Rise" event also makes the task more difficult.

Table 5 shows the performance of event extraction, which is the final output of our system. These event sets were prepared apart from relation extraction training/test set, and were also manually labeled. The news event set consists of 1152 sentences collected from news articles. The blog event set consists of 813 sentences collected from blog articles. The tweet event set consists of 810 sentences collected from Twitter.

As mentioned before, event extraction requires POS tagging, named entity recognition, dependency parsing and relation extraction as preliminary steps. The result in Table 5 is the aggregate performance of all the modules in the pipeline. The experiment was conducted using the Hybrid Model for simplicity's sake. Because other modules than relation extraction were heavily trained on news articles, the aggregate performance is particularly high in news domain. Tweets are especially difficult to process, being most ungrammatical and abundant in tweet-specific acronyms and abbreviation, which explains the low performance. As our ultimate task is to build a social media monitoring system, we tried to maintain high precision in all cases, in effort to avoid presenting incorrect events to users.

5. Conclusion and Future Work

In this paper, we described a social event extraction using structured SVM. We defined 31 events to address social issues and broke them down to 130 binary relations. We used structured SVM to perform multiclass classification. The Event Detector was developed in the process to adapt the system to the event-extraction task. Using various training methods, we compared the performance of different models and showed that a hybrid model is generally most competent. Our approach, combined with other modules, showed 65.6 F1 for event extraction from news articles, 43.4 F1 for blog articles, 33.4 F1 for tweets.

This study was conducted assuming that relations in the same sentence are independent of one another. But this assumption often does not hold. For future work, we intend to use Markov random field [19] to fully express the dependencies between relations in the same sentence. We also plan to expand the range of social events, possibly including defense/military domain. And to further increase the precision, sentences that contain hypothesis, demand, question, or speculation need to be filtered out. And finally we plan to use a bootstrapping method or distant supervision to automatically construct labeled data to reduce manual labor.

References

[1] Recorded Future, https://www.recordedfuture.com/
[2] Social Wisdom, http://wisdom.daumsoft.com/wisdom/kr
[3] D. Roth and W. Yih, “Probabilistic reasoning for entity & relation recognition,” Proc. of COLING, 2002
[4] N. Kambhatla, “Combining lexical, syntactic and semantic features with maximum entropy models,” Proc. of ACL, 2004
[5] D. Zelenko, C. Aone, and A. Richardella, “Kernel methods for relation extraction,” JMLR, vol.3, pp.1083-1106, 2003
[6] A. Culotta and J. Sorensen, “Dependency tree kernels for relation extraction,” Proc. of ACL, 2004
[7] S. Zhao and R. Grishman, “Extracting relations with integrated information using kernel methods,” Procs. of ACL, 2005
[8] G. Zhou and Q. Zhu, “Kernel-based semantic relation detection and classification via enriched parse tree structure,” Journal of Computer Science and Technology, vol.26, no.1, pp.45-56, 2011
[9] D. Ravichandran and D. Hovy, “Learning surface text patterns for a question answering system,” Procs. of ACL, 2002
[10] P. Pantel and M. Pennacchiotti, “Espresso: leveraging generic patterns for automatically harvesting semantic relations,” Proc. of ACL, 2006
[11] M. Banko, M. Cafarella, S. Soderland, M. Broadhead and O. Etzioni, “Open information extraction from the web,” Proc. of IJCAI, 2007
[12] M. Banko and O. Etzioni, “The tradeoffs between open and traditional relation extraction,” Proc. of ACL, 2008
[13] C. Wang, J. Fan, A. Kalyanpur and D. Gondek, “Relation extraction with relation topics,” Proc. of EMNLP, 2011
[14] C. Lee, Y. Hwang and M. Jang, “Fine-grained named entity recognition and relation extraction for question answering,” Proc. of SIGIR, 2007
[15] I. Tsochantaridis, T. Joachims and Y. Altun, “Support vector machine learning for interdependent and structured output spaces,” Proc. of ICML, 2004
[16] C. Lee and M. Jang, “Fast training of structured SVM using fixed-threshold sequential minimal optimization,” ETRI Journal, vol.31, no.2, pp.121-128, 2009
[17] T. Joachims, T. Finley and C. Yu, “Cutting-plane training of structural SVMs,” Machine Learning Journal, vol.77, no.1, pp.27-59, 2009
[18] S. Shalev-Shwaratz, Y. Singer and N. Srebro, “Pegasos: Primal estimated sub-gradient solver for SVM,” Procs. of ICML, 2007
[19] T. Finley and T. Joachims, “Training structural SVMs when exact inference is intractable,” Proc. of ICML, 2008
[20] C. Lee and M. Jang, “A prior model of structural SVMs for domain adaptation,” ETRI Journal, vol.33, no.5, pp.712-719, 2011