Action Extraction Based on Open IE

Zhichao Wang¹, Lian Duan¹, Jiuming Huang¹, Aiping Li¹, Zhonghe He², Jianjing Guo²

¹ Department of Computer, National University of Defense Technology, Changsha, Hunan, 410073, China
² Hunan Xinghan Data and AI Technology Co., Ltd, Changsha, Hunan, 410073, China
wangzhichao15@nudt.edu.cn

Abstract. A centre challenge in enhancing usage of natural language text is most sentences are barely clearly shows useful information for relation extraction, event extraction, etc. However, information cross sentences or even cross documents bounded together may produce better results. We bring action extraction based on open IE results to extract actions as meta results. It not only presents what an entity did and will do, but also provides an important foundation of producing conclusive results by using statistics and deduction. By using action extraction, we dramatically generate vast mention pairs we called weak relation which means the pair of mentions exist in the same action. This paper is focus on constructing a knowledge base not only with relations, but also actions so that we can do more work in future.

1. Introduction and Related Work

Since Open IE [1] comes out, many researchers has tried to use it on Information Extraction to improve utilization of natural, unstructured language. Most of open IE related works are focus on extracting entities and relations from the web [2]. There are many papers and systems are focus on improving performance [3–6] of open IE. The occurrence of open IE shows possibility of doing information extraction on open domain data. And many researchers are using open IE as intermediate results for specific tasks, such as Relation Extraction [7,8]. However, the utilization of natural language is still small. For example, open IE on relation Extraction can only use about 10% of tuples (1 million of 11.3 million [1]) to extract given relations. The rest of tuples are useless because of the limit of relations or lack of evidence to show relation between entities in this sentence. The reason of this kind of waste is because a sentence mostly shows one-sided or just a hint of the relations between two entities. Therefore, it is hard to correctly assert relation in one sentence. But, like detective solving case, they can deduce relations between two people based on multi "evidences". Therefore, we are trying to focus on cross-sentence and cross-document information extraction. The extremely urgent procedure we need to do, is to find a structure to store “evidences” for afterward offline process which can combine evidences for analysis and deduction.

When we analysis Stanford CoreNLP open IE [5] results, we found the meaningful tuples(evidences) are mostly described actions of Named Entity. The world is described by a set of temporally qualified actions outlining what is known about the past, present, and future. [9] In this paper, we bring the conception of action into natural language processing field. There are at least two major features of actions which are meaningful for natural language processing:

- Actions that involve at least one Named Entity or Named Entity Mention;
- Actions that describe a dynamic perspective [9] of the world (static perspective actions can
be defined as property of a named Entity. E.g., Donald Trump is president of USA.)

Additionally, actions would be more meaningful if contains time and location when and where they happened.

Action in NLP is different from event. Action focus on every kinds of dynamic perspective of acts while Event only focus on specific acts which are pre-defined as “event type” [10], such as Life, Movement, Transaction, Business, etc. Event Extraction pays attention on the most major events of entities. Therefore, EE ignored most of actions which are not that important. (e.g. Alice plays basketball with Bob in NYU in Sunday).

This paper also brings a way to store action in knowledge base. Nowadays, people are tend to use RDF – the smallest data unit of World Wild Web [11] as scale for knowledge base construction. In the most fundamental form, RDF involves mostly two entities (as subject and object) and one description(predicate) to describe connection between subject and object. But in action, it will involve many arguments which are all requisite for describing an action. In a RDF-based knowledge base, we defined a new type of virtual nodes to describe action as a named Entity so that all arguments would be the relation or property of Virtual Action Node.

Actions Extraction is pretty much like Semantic Role Labelling(SRL) to label different arguments associated with verbs [12]. However, SRL have distinct roles for different predicate while Action Extraction using the common roles to represent different predicates. Open IE is closely referred to SRL [13,14] to extract arguments from sentences. However, SRL systems traditionally rely on syntactic parsers, which makes them susceptible to parser errors and substantially slower than Open IE systems [7]. Still, the connection between open IE and SRL are tied. In 2010, Janara Christensen [15] represents a method to improve Open IE performance by using SRL. After observation of Open IE results, we found SRL arguments are well extracted into binary tuples which connected with same predicate. Therefore, we try to reverse the application by using open IE results to generate SRL-liked structure action.

Action is a common description so that we divided their arguments as follows:

- Subject, the sender of action
- Predicate, the concrete description of action, usually a verb, and can be both passive and active.
- Object, the receiver of action
- Time, the specific time that action connected
- Location, the specific location that action involves
- Other Argument, arguments which cannot be classified as above but useful

Figure 1 demonstrates an example of action storage structure in Knowledge Base. As we can see, one action can have multi subjects, objects, time, location.
Also, in this paper, we bring OAE, (OpenIE-based Action Extractor), an action extractor based on open IE technique of the Stanford CoreNLP.

Natural Language Processing ToolKit [16], which can detect action and classify action arguments to those six arguments above from unstructured text.

Section 1 introduces motivation of this paper and analyzes previous work. In the remainder of the paper is organized as follows. Section 2 represents one novel way of processing natural language, especially action extraction. Section 3 reports on our experimental results and analysis. Section 4 concludes with a summary and discussion of future work.

2. OpenIE-based Action Extraction

The key idea of OAE is using the good constructor of open IE results which contains action arguments in a triple tuple way. CoreNLP provides the meta results for extracting and classifying action arguments into six kinds arguments: subject, predicate, object, time, location, other. By exploiting open IE’s features, we can easily generate a domain-independent extractor for actions. OAE has five steps to extract and store actions.

2.1. Tuple Generator

The generator generates open IE results for natural, unstructured language by using Stanford CoreNLP Toolkits. We opened Anaphora Resolution in paragraph level for replacing subject and object of tuple which is pronoun to improve utilization rate of texts. The results are expressed as below:

$$(t^1_1, t^1_2, t^1_3, \ldots, t^1_n)$$

In (1), subscript of t is group id, superscript of t is tuple id in group.

Knowledge base tend to construct an entity-level base for storing entity relations and properties whereas the tuple arguments are not simply entity but also contains modifier of entity. Therefore, we reduce arguments to basic core word when the core word is a named entity, and regard modifiers as title, occupation or dropped.

2.2. Tuple Grouper

Gabor Angeli [5] mentioned that open IE in CoreNLP is not simply representing tuples with independent clauses, but also tried to shorten arguments more likely to be useful for downstream applications by using natural logic framework. Therefore, open IE represents all possible correct
results for arguments with various level of information. We choose four features to show if two tuples described the same argument:

- Core word offset of tuple subject
- Core word offset of tuple predicate
- Core word offset of tuple object
- Last word offset of tuple predicate if it is preposition

So, the tuples describing same arguments will be grouped together (results are shown in Table 1).

| Group ID | Tuple ID | Candidates |
|----------|----------|------------|
| 1        | 1        | <Donald-0, attend-5, meeting-8> |
| 1        | 2        | <Donald-0, attend-5, great-7 meeting-8> |
| 2        | 1        | <his-2 wife-4, attend-5, meeting-8> |
| 2        | 2        | <his-2 wife-4, attend-5, great-7 meeting-8> |
| 2        | 3        | <his-2 beautiful-3 wife-4, attend-5, meeting-8> |
| 2        | 4        | <his-2 beautiful-3 wife-4, attend-5, great-7 meeting-8> |

The way to find core word is using dependency parse to judge each word if its governor is inside the same tuple arguments. Figure 2 shows that Donald is governor of wife and is not part of tuple subject in \( t_3 \), so that wife is the core word. The tuple grouping result obtaining an accuracy of 100% in manually labeled 589 sentences which contains at least one action.

2.3. Tuple Selector

Tuples in the same group describes the same arguments with different level of information. Therefore, we need to choose one tuple for each group which is best correspond to the principle: On the premise of keeping semantic integrity, the shorter one is the better one.

Since we already processed tuple arguments which core word is Named Entity, we shall concentrate on the other side. After our analysis of the result of open IE, we found that named entities (Table 2) are preferred to be kept, and proper nouns (NNP) and numeric words (Table 3) are also worth saving. Include means amount of words the best result contained more than the closest shorter one than best, whereas exclude shows the longer one contained words need to drop to produce best result. Besides, we found that tuple with prepositional phrase contains more valuable information than the ones without. Thus, the method we use is to keep all numeric words, named entities, proper nouns and prepositional phrase while as short as possible.

| Group ID | Tuple ID | Candidates |
|----------|----------|------------|
| 1        | 1        | <Donald-0, attend-5, meeting-8> |
| 1        | 2        | <Donald-0, attend-5, great-7 meeting-8> |
| 2        | 1        | <his-2 wife-4, attend-5, meeting-8> |
| 2        | 2        | <his-2 wife-4, attend-5, great-7 meeting-8> |
| 2        | 3        | <his-2 beautiful-3 wife-4, attend-5, meeting-8> |
| 2        | 4        | <his-2 beautiful-3 wife-4, attend-5, great-7 meeting-8> |

Table 2: Named Entity Type of words tend to include or exclude for getting best answers

|          | person | location | organization | time |
|----------|--------|----------|--------------|------|
| include  | 37     | 52       | 43           | 17   |
| exclude  | 4      | 7        | 4            | 0    |
Table 3: Part-of-Speech of words tend to include or exclude for getting best answers

|     | nnp | num | adj  | adv | verb | nn  | in(to) | other |
|-----|-----|-----|------|-----|------|-----|--------|-------|
| include | 28  | 7   | 24   | 3   | 2    | 66  | 69     | 9     |
| exclude | 1   | 0   | 50   | 11  | 11   | 6   | 6      | 4     |

We demonstrate a novel method to retrieve the best tuple in one tuple group by constructing multi-tree to present tuple object hierarchical overlapping relationships among tuples after shortening subject and predicate:

**Step1.** Construct a multi-tree of tuples in one group while fitting the principle below.

\[ t^i_{child} = t^k_{child} \text{ if and only if } \]
\[ \text{overlap}(t^i_{object}, t^k_{object}) = t^k_{object} \text{ and overlap}(t^i_{object}, t^k_{object}) \neq t^k_{object}(x \neq j, k) \]

Overlap is Longest Common Subsequence \[17\] algorithm to find overlapping words of two string. The principle ensures that the inclusion relations are hierarchically.

**Step2.** Remove edges which lead the result of dropping key words like word.pos = nnp, word.nerType ≠ O, word.lemma contains digit and word is preposition.

**Step3.** Get all leaf node tuples as candidate results of the best result. Mostly, candidate result set contains only one result. If not, we choose the shortest one as the best result.

2.4. Action Grouper
In this step, we are going to cluster tuples into action level which describe the same action. The results are easily divided to different actions by using core word offset of predicate, because an action is about the arguments with same predicate.

2.5. Action Argument Classifier
The decisive step for action extraction is action argument identification to divide tuple arguments into action arguments as subject, predicate, object, time, location, and other.

By using Open IE as intermediate result, we found that tuple subject is absolutely action subject, and predicate is absolutely action predicate. So, we only need to classify tuple object to action arguments.

Action arguments are partly depended on dependency parse results. We assume tuple object classification is about object core word classification and other words in tuple object are modifier of core word. Therefore, we build an action argument classifier, which predicates which kind of argument a tuple object is. We use manually labeled 600 action tuples mentioned before to get argument type from each tuple.

For learning, we use Support Vector Machines \[18\]. The features used include: which preposition the predicate ends with, dependency type from preposition to, dependency type from the core word to, tuple object core word dependency. We compare the classifier to a basic method: use only dependency of tuple object core word to decide which type of argument object be.

3. Evaluation
To evaluate the ability of our approach to extract action, we analysis the outcome of OAE with 4 aspects below:

**Argument selection in Group:** Our method for argument selection in group in our manually labeled data bring the result of 84.4% accuracy (Figure 3) and steadily remains around 84.4% with randomly selected actions in size of 100, 200, 300. Comparing method selected results to labeled correct results, we found that most of the wrong results are due to few adjectives having semantic shift impact being removed by mistake. For example, “Alice became a rich man yesterday.” brings results of \(< Alice, become, rich man >\) and \(< Alice, become, man >\), the meaning of sentence has semantic shift impact while removing the adjective “rich”.


Argument classification: argument types are subject, predicate, object, time, location, other. For tuple object argument classification, we randomly select 500 actions as training data and 100 actions left as test set. results are shown in Table 3. We can see the classifier obtains a 13% increase in F1 score average. Also, the figure shows SVM classifier is better than dependency classifier in most arguments by increasing their recall figure as well as their accuracy. We assume it’s because that the pattern for classifying different argument is different features combined together other than dependency itself.

Weak Relations: entities bounded in the same action could represent connection between them but cannot easily fill into pre-defined relation slot. However, we still define it as weak relation as “evidence” of deducing relations between two entities afterward with enough statistics and deduction evidences. Table 4 presents that result gets 1142 mention pairs which are bounded in one action while Stanford’s Distantly-Supervised Slot-Filling System [19] only detected 84 relations. It reveals the possibility of connecting entities without pre-defined relation. For example, we don’t know which kind of relation Alice and Bob has, but we know they played basketball together, so that we can bound them with this action and generate a path between them.

4. Conclusion
In this paper, we introduce the definition of action in NLP and the definition of action extraction. Also, we bring a method of action extraction by using open IE results (OAE). Different from other applications of using open IE, we truly improve utilization rate of unstructured text by using action extraction because of extra tuples presenting action being extracted and stored.

This paper also shows advantage of storing action in knowledge base. Knowledge base tried to store everything. However, because of knowledge basic structure and the KB storage and query structure, knowledge graph required a more structured way to store information. We bring virtual node “action” to connect all action arguments, so that entities involved in one action can be connected by this node as weak relation, a co-occurrence relation based on semantic information. Our statistics shows that action could connect 1142 entity pairs, and action data can be dependency of deduction, ontology generation, etc.
Table 4 results of argument classification

|                      | subject | predicate | object | time | location | average |
|----------------------|---------|-----------|--------|------|----------|---------|
| correct result number| 121     | 100       | 90     | 15   | 34       | -       |
| dependency result number | 107  | 100       | 102    | 15   | 25       | -       |
| accuracy             | 100     | 100       | 72.55  | 46.67| 66.67    | 77.18   |
| recall               | 88.43   | 100       | 85.06  | 46.67| 48.5     | 73.73   |
| F1                   | 93.9    | 100       | 78.3   | 46.67| 56.14    | 75.00   |
| svm with multiple features result number | 117  | 100       | 96     | 16   | 27       | -       |
| accuracy             | 94.02   | 100       | 89.58  | 75.00| 77.78    | 87.28   |
| recall               | 90.91   | 100       | 95.56  | 80.00| 84.00    | 90.09   |
| F1                   | 92.44   | 100       | 92.47  | 77.42| 80.77    | 88.62   |

In the future, we are going to seek the method of judging semantic impact of each word in one sentence so that we can get better outcomes from argument selection in group. We will also focus on using action extraction as meta data to deduce final relation between two entities after we finish our cross-document entity linking research.

Acknowledgements
This work is supported by The National Key Research and Development Program of China (2016QY03D0601, 2016QY03D0603) and National Natural Science Foundation of China (No.61502517, No.61672020, No.61662069).

References
[1] Michele B, Cafarella M J, Oren E. Open information extraction from the web: WO, WO 2008121144 A1[P]. 2008.
[2] Gupta S, Manning C. Improved Pattern Learning for Bootstrapped Entity Extraction[C]// Eighteenth Conference on Computational Natural Language Learning. 2014:98-108.
[3] Bast H, Haussmann E. More Informative Open Information Extraction via Simple Inference[C]// European Conference on Ir Research on Advances in Information Retrieval. Springer-Verlag New York, Inc. 2014:585-590.
[4] Etzioni O, Fader A, Christensen J, et al. Open Information Extraction: The Second Generation.[C]// IJCAI 2011, Proceedings of the, International Joint Conference on Artificial Intelligence, Barcelona, Catalonia, Spain, July. DBLP, 2012:3-10.
[5] Angeli G, Premkumar M J, Manning C D. Leveraging linguistic structure for open domain information extraction[C]//Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics (ACL 2015). 2015.
[6] Wu F, Weld D S. Open information extraction using Wikipedia[C]// ACL 2010, Proceedings of the, Meeting of the Association for Computatonal Linguistics, July 11-16, 2010, Uppsala, Sweden. DBLP, 2013:118-127.
[7] Fader A, Soderland S, Etzioni O. Identifying relations for open information extraction[C]// Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, 2011:1535-1545.
[8] Soderland S, Roof B, Qin B, et al. Adapting Open Information Extraction to Domain-Specific Relations[J]. Ai Magazine, 2010, 31(3):93-102.
[9] Allen J F. Towards a general theory of action and time[M]. Elsevier Science Publishers Ltd. 1984.
[10] Aguilar J, Beller C, Mcnamee P, et al. A Comparison of the Events and Relations Across ACE, ERE, TAC-KBP, and FrameNet Annotation Standards[C]// The Workshop on Events: Definition. 2014:45-53.
[11] Punnoose R, Crainiceanu A, Rapp D. Rya:a scalable RDF triple store for the clouds[C]// International Workshop on Cloud Intelligence. ACM, 2012:1-8.
[12] Palmer M, Gildea D, Kingsbury P. The Proposition Bank: An Annotated Corpus of Semantic Roles[J]. Computational Linguistics, 2005, 31(1):71-106.
[13] Punyakanok V, Roth D, Yih W T. Wen tau Yih. 2008. The importance of syntactic parsing and inference in semantic role labeling[J]. Computational Linguistics, 2008, 34(2):257-287.
[14] Haghighi A, Toutanova K, Manning C D. A joint model for semantic role labeling[J]. Computational Linguistics, 2005, 34(1):161-191.
[15] Christensen J, Soderland S, Etzioni O. Semantic role labeling for open information extraction[C]// NAACL Hlt 2010 First International Workshop on Formalisms and Methodology for Learning by Reading. Association for Computational Linguistics, 2010:52-60.
[16] Manning C D, Surdeanu M, Bauer J, et al. The Stanford CoreNLP Natural Language Processing Toolkit[C]// Meeting of the Association for Computational Linguistics: System Demonstrations. 2014.
[17] Hirschberg, Daniel S. Algorithms for the Longest Common Subsequence Problem[J]. Journal of the Acm, 1977, 24(4):664-675.
[18] Tong S, Koller D. Support vector machine active learning with applications to text classification[J]. Journal of Machine Learning Research, 2002, 2(1):45-66.
[19] Surdeanu M, Gupta S, Bauer J, et al. Stanford's Distantly-Supervised Slot-Filling System[C]//TAC. 2011.