A Survey of Open Domain Event Extraction

Zecheng Zhang
Department of Computer Science
University of Illinois at Urbana-Champaign
zzhan147@illinois.edu

Yuncheng Wu
Department of Computer Science
University of Illinois at Urbana-Champaign
ywu101@illinois.edu

Zesheng Wang
Department of Computer Science
University of Illinois at Urbana-Champaign
zwang180@illinois.edu

ABSTRACT

The development of the Internet and social media have instigated the increasing of text information communication and sharing. Tremendous volume of text data is generated every second. To mine useful and structured knowledge from the unstructured text corpus, many information extraction systems have been created. One of the crucial methods of those systems is the open domain event extraction, which aims at extracting meaningful event information without any predefined domain assumption. Combining with some recent promising techniques including Question Answer pairing, entity linking, entity coreference and deep learning, the result of open domain extraction seems to be improved. In this survey, we will first briefly give introduction to the pipeline of event extraction. Then we will give relatively detailed description on recent promising open domain event extraction approaches including examination of some recent papers.

1. INTRODUCTION

The amount of text data has been rapidly increased due to the spread of Internet and mobile access in recent years. To get interesting, representative and human-interpretable knowledge from those text data, many data mining techniques have been created. Most text data from social media and the Internet is unstructured in the machine view and merely composed by some human readable natural languages [1]. Therefore, it is difficult for machine or computers to learn and extract the underlying structures from those data automatically. These are the main difficulties for some related applications such as information retrieval system and information extraction system which can provide useful service for human beings [1].

Event extraction is one of the most useful techniques for information retrieval and information extraction systems. It has been studied for long time but still remains a challenging task. The purpose of extraction is to detect type of events and extract arguments with difference roles from human-readable text automatically by machine [6]. For example, given a sentence Students in Siebel Center destroyed a computer last month, as given in Figure 1, an event extraction system need to detect the event Destroy triggered by *destroyed* and can identify corresponding roles of arguments in the sentence: students (Role=Destroyer), Siebel Center (Role=Location), computer (Role=Target) and last month (Role=Date). Good event type and argument roles extraction can significantly help information retrieval and information extraction systems.

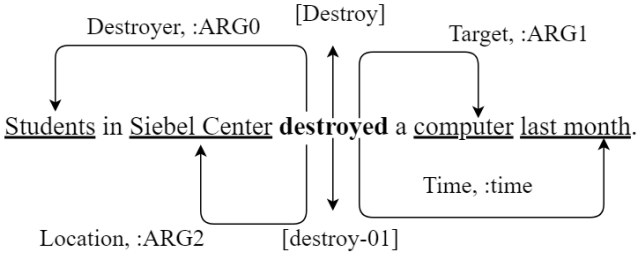


Figure 1: An Event Mention example: **destroyed** triggers a Destroy event with four arguments.

Nevertheless, automated event extraction is still a challenging task for machine. It is relatively difficult to assign correct types for unseen event mentions and corresponding roles to arguments. In addition, many event have complex and nested structures [2]. This gives the automated event extraction even more obstructions. For example, sometimes a crisis event can cause an investigate event and then result in an announce event which difficulties in identifying exact structure [2]. Even in fields such as biology, some biomolecular events also have nested and complex structures. Many methods or approaches have been used for event extraction such as using the predefined event types and schema. But these methods exhibit very low coverage on typing trigger words and arguments due to the complex and nested structure of event types and unseen event mention.

In recent years, some new methods have emerged and have shown promising solution for automated information extraction for massive text corpus such as using the reading comprehension and question answering, deep learning techniques (notably, convolutional neural network). More details of these recent approaches will be introduced and examined in section 4. In the meantime, we are also exploring a new direction which relies on previous work related to graph mining on heterogeneous information network. In order to properly formulate the network, we survey literatures in the field of entity linking/grounding, co-reference resolution and clustering to help us construct quality and dense information network from the raw data, such it meets the requirements of performing an open domain event extraction. Related details will be introduced in section 5 and section 6.

Throughout this survey paper, in section 2 we would give some introduction on types of event extraction on different

aspects. Then, in section 3 we will give introduction on some existing knowledge base for event extraction and techniques automatically generating high quality event training data for event extractions. In addition, in section 4, 5 and 6 there would be some more specific evaluations for various promising event extraction approaches and some useful techniques for open domain event extraction. Those approaches and techniques discussed in section 4, 5 and 6 mainly are some contemporary promising ones such as utilization of question answering pair and some deep learning techniques in open domain event extraction tasks. Finally, we would summarize on event extraction and give summarization of this survey paper.

2. TYPES OF EVENT EXTRACTION

There are several criteria to define the types of event extraction. Here we summarize two main kinds of criteria on distinguishing types of event extraction. First, discussed in section 2.1, the criteria is to separate the event extraction by objective, whether the event extraction is aimed to be applied on open domain or on specific domains. Second, the distinguishing criteria is based on different approaches which is summarized in section 2.2. In that section, we categorize event extraction approaches into three ones, data-driven approaches, semantic-driven approaches and hybrid-driven approaches [1].

2.1 Event Extraction Types by Approaches

Event extraction techniques have involved utilization of various approaches and models. Nevertheless, some event extraction methods have similar heuristics and approaches on their modeling which we can cluster similar ones together. Thus, in this section we mainly classify types of event extraction into three main approaches. The first one is the approach using quantitative statistical models which associated with more closely to the statistical pattern of text data; also as described in [1] this approach will usually be called as data-driven approach. The second one is an approach semantic-driven approach incorporating models derived from linguistic, semantic and lexical knowledge. The last one is a hybrid-driven extraction approach that usually combines the first and second approaches [1].

Data-driven approaches have been used widely in the field of event extraction. As what we call the approach as data-driven, statistical models usually are adopted in trying to find the data-based or global-based patterns of event schema in the event extraction. For example, in the paper Discovering Volatile Events in Your Neighborhood Local-area Topic Extraction from Blog Entries [5], they first treat event extraction as some topic and subtopic extraction and then utilize statistical methods such as the z score significance test to rank those topics or subtopics [5]. These are the kinds of purely data-driven approaches. Even if they have been widely used, it still has some limitations. For example, purely statistical methods cannot incorporate information between some relations and might lose semantic meaning even the model provides some good statistical result [1]. In summary, data-driven approach usually provides with a relatively global perspective on the data on event extraction but might lose some locality information such as part of semantic meaning in sentences.

Besides the data-driven approach, semantic-driven approach is also usually used in event extraction tasks. Compared to the statistical methods used in the data-driven approach which might lose some semantic meanings, semantic-driven approach is almost based on lexical knowledge and more human interpretable methods that focuses most on relative local semantic pattern during the event extraction that retain most semantic meanings. For example, in the paper High-precision Biological Event Extraction with a Concept Recognizer [3], some related ontologies [13] are used to support the event extractions. The use of those syntactic and semantic elements can be well human-interpretable and need less data for referencing or training compared to the data-driven approaches [1]. However, many prior knowledge and efforts are needed to specify those syntactic and semantic elements which can be relatively expensive and time-consuming [1]. Also, some local syntactic and semantic patterns cannot be incorporated well globally after first specifying, resulting in other demands for further tuning of these syntactic and semantic patterns [1]. Furthermore, in some situations, it would be unrealistic or difficult to generate those semantic and syntactic patterns in low cost.

After the introduction on the data-driven approaches and semantic-driven approaches, we can find they have different advantages and disadvantages. Since their advantages and disadvantages are in different area, it is possible to combine them and mutually enhance the final result of event extraction. Thus, the third approach comes which combine the advantages from both data-driven and semantic-driven approaches and usually provide better event extraction results than those just using a single kind of approach. According to its attributes we called it hybrid-driven event extraction approach [1]. For example, in the paper Zero-Shot Transfer Learning for Event Extraction, they utilize some ontologies such as FrameNet but also combine some statistical models which generate good event extraction results. In conclusion, contemporary methods usually adopt the third kind of approach which we will give more specific examples and evaluations in later sections.

2.2 Event Extraction Types by Objectives

Event extraction is usually supposed to be applied into different fields such as medical, biological, finance and social media. Different objectives of tasks on event extraction result into different utilization and creation of event extraction techniques and approaches. Thus, in this section we mainly categorize the event extraction into two types, the open domain and specific domain event extraction.

Many event extractions are applied into some specific realm such as biology, finance and medicine. In these realms, the purpose for event extraction of text data usually closely related to their own profession terminology. Therefore, in many specific domain event extraction, some reference ontologies or knowledge bases are widely used to help with event extraction. In some event extraction literatures they use corresponding ontologies. For example, in High-precision Biological Event Extraction with a Concept Recognizer, they use a system called OpenDMAP which is an ontology-driven and biomedical concept analysis system combining with several community-consensus ontologies to support their approached on biological event detection and argument identification [3]. Existing ontologies, specific domain knowledge bases and semantic models can significantly support

the event extraction. Even if there will be some changes or some new event types created in specific domains, it would be relatively easy to incorporate these changes into event extraction models such as making modifications on corresponding ontologies or semantic models; thus, they can still provide significant support on event extraction.

Compared to objective of applying event extraction on specific domains which have referencing ontologies and specific helpful knowledge bases, many projects aim to have event extraction on non-specific or open domains. For example, usually event extractions on social medias and massive noisy text corpus are usually categorized as open domains extraction. Even if some knowledge bases can also be helpful such as Wikipedia [11]. But since the text we want to apply the open domain event extraction is noisy, complex and even have no associations to any knowledge bases, these knowledge bases cannot display the same crucial roles as what they do in specific domains. For instance, in the literature Open Domain Event Extraction from Twitter, they do not use specific knowledge bases or ontologies since tweets in twitter are noisy and including a great amount terminology non-existing in all knowledge bases [4]. In conclusion, lack of existing reference ontologies and a large amount of complex underlying event types and schema cause the open domain event extraction from large and noisy text corpus more challenging but can be more useful in the future.

Because of rapid increasing number of massive and relative noisy text data such as posts from social media and reviews from numerous website, event extraction on these open fields become more interesting which specifically need the development of open domain event extraction techniques. In the following sections of this survey, we will give some introduction and evaluation on several contemporary approaches useful or promising for open domain event extraction.

3. DATA AND ONTOLOGIES

Increasing number of text data has been instigating the development of event extraction techniques. Many models of event extraction, specifically for open domain ones, are based on supervised learning from small hand-labeled data. The hand-labeled data has three disadvantages: expensive to produce, low coverage of event types, and limited in size. Thus, supervised models for event extraction that are trained on hand-labeled data are hard to extract large scale of events for knowledge base population. To exploit large amount of existing unlabeled text data, people proposed methods to automatically generate labeled data for event extraction from knowledge base. In this section, we will at first introduce resources of useful event extraction data and ontologies. Then we will talk about two approaches to generate labeled data for event extraction. The first one is generating data by identifying key arguments and trigger words and the second one is generating data only by identifying key arguments.

3.1 Resources of Data for Event Extraction

There are many existing knowledge bases or ontologies useful for open domain event extraction task; to generate labeled data for open domain event extraction, not only the word knowledge is necessary, but also need the linguistic or semantic knowledge. Here are some well-known ontologies or knowledge bases used to generate or used as labeled data

Table 1: Wikipedia Multilingual statistics (Top 10 in 2018)

Language	Articles	Growth(year)
English	2,567,509	+26%
German	808,044	+25%
French	709,312	+26%
Polish	539,688	+26%
Japanese	523,639	+25%
Italian	499,234	+41%
Dutch	481,064	+31%
Portuguese	429,730	+49%
Spanish	402,430	+42%
Russian	318,850	+55%

for event extraction:

Remedy Corp was sold to BMC Software as the Service Management Business Unit in 2004.
company acquired | acquiring company | divisions formed | date

Figure 2: An example sentence from Wikipedia.

3.1.1 *FreeBase:*

Freebase is a semantic knowledge base. It uses Compound Value Types (CVT) to combine multiple values into a single value. As shown in Figure 2 and Figure 3, *business.acquisition* is the type of the CVT entry. There are many instances of *business.acquisition*, so this instance has a unique id *m.07bh4j7*. *Company_acquired*, *acquiring_company*, *data* and *divisions _formed* are roles of *business.acquisition* CVTs. *Remedy Corp*, *BMC Software*, *2004* and *Service Management Business Unit* are values of the instances. To exploit FreeBase as a resource to generate labeled data, CVTs are usually treated as events, types of CVTs are treated as event types, CVT instances are treated as event instances, values in CVTs are treated as arguments in events, and roles of CVTs are treated as the roles of arguments in the event.

id	company_ acquired	acquiring_ company	date	divisions_ formed
m.07bh4j7	Remedy Corp	BMC Software	2004	Service Management Business Unit

Figure 3: the corresponding CVT entry of the sentence in Figure 2 in FreeBase.

3.1.2 *FrameNet:*

FrameNet is a linguistic resource storing information about lexical and predicate argument semantics [23]. Each frame of FrameNet can be taken as a semantic frame of a type of events [24]. Each frame has a set of lemmas with POS tags that can evoke the frame which are called Lexical Units. For example, in the sentence shown below, *bake.v* is a Lexical Unit of *Cooking_creation* frame in FrameNet. Such frame can then be mapped to *Cooking_creation* in FreeBase, and then be used to label text data.

Michelle baked her mother a cake for her birthday.

3.1.3 *Wikipedia:*

Wikipedia, as one of the largest, semi-structured and wide-coverage knowledge basse [14], can be helpful for entity link-

ing and thus support the entity extraction. As Table 1 shows, one of the largest advantages for Wikipedia is that it is a multilingual based knowledge base that has hundreds of languages of articles and each has a relatively rapid increasing numbers. Thus, some techniques such as wikification can be extremely helpful for event extraction that can link same meaning entity mentions to specific one and then enhance the open domain extraction results. More details about using Wikipedia as a helper method for event extraction will be discussed in section 5.

3.2 Generate Labeled Data for Event Extraction Automatically

Many current promising methods on open domain extraction usually need large amount of labeled data such as some deep learning approaches (will be discussed in section 4). Following are some recent approaches to generate labeled data for event extraction automatically and with relatively high accuracy which can significantly help the open domain event extraction in some models.

3.2.1 Generate labeled data for event extraction by identifying trigger words and key arguments [6]

The objective of generating labeled data for event extraction in this approach is to generate data which involves labeling triggers, event types, arguments and their roles. For example, Figure 4 shows a labeled event mention example for event extraction. Word *threw* is the trigger of the event whose type is *Attack*. Words underlined in the sentence are arguments of this event.

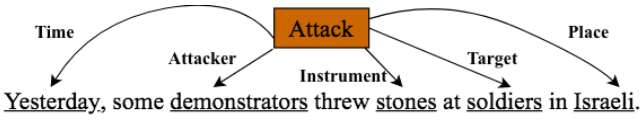


Figure 4: Another example of labeled sentence for event extraction.

Existing work on distant supervision for relation extraction assumes that if two entities have a relationship in a known knowledge base, then all sentences that mention these two entities will express that relationship in some way. However, such distant supervision does not work well on event extraction, because it has two problems. The first one is that triggers are not given out in existing knowledge bases. The second one is that arguments for a specific event instance are usually mentioned in multiple sentences, so simply using all arguments in the knowledge base to label back sentences will generate not enough high quality sentences as training samples.

To resolve problems in applying distant supervision on event extraction, there is an approach jointly uses world knowledge (FreeBase) and linguistic knowledge (FrameNet). Here is the pipeline to generate labeled data in this way:

- Select key arguments.
- Only use key arguments to label events and extract trigger words.
- Use external linguistic knowledge resource (FrameNet) to filter noisy trigger words and expand more triggers.
- Use Soft Distant Supervision to automatically label training data.

The first step of the entire pipeline is key argument extraction. The paper[6] defines key argument as argument that plays indispensable roles in an event and serves as vital clues when distinguishing different events. For example, argument *spouses* are key arguments in a *marriage* event rather than *time* and *location*. To measure the importance of an argument, the paper[6] introduces a measurement called **Key Rate**, which is combined by two terms: **Role Saliency** and **Event Relevance**. **Role saliency** reflects the saliency of an argument to represent a specific event instance of a given event type. In other words, given an event type, an argument A is more salient than another argument B if one tends to use an argument A to distinguish one event instance. Event relevance reflects the ability that an argument can be used to discriminate different event types. In other words, if an argument A only occurs in a specific event type, the argument A has high event relevance. The general idea to extract key argument is pretty straightforward: for each event type, compute key rate for all arguments under that type. Then for each type, select top K arguments as key arguments.

The second step is trigger word detection. The paper[6] used Stanford CoreNLP tool to convert raw Wikipedia text data to sequence of sentences with NLP annotations and selects sentences that contain all key arguments of an event instance in FreeBase as sentences expressing corresponding events. Then the paper[6] used these selected sentences to detect trigger words. The general idea is similar to TF-IDF. A verb tends to express an occurrence of an event in a sentence. If a verb occurs more time than other verbs in the labeled sentence of one event type, the verb tends to trigger this type of event. However, if a verb occurs in sentences of every event types, like *are*, the verb will have lower probability to trigger specific type of event. In this way, some trigger words can be extracted for each event type in this step.

The third step is trigger words filtering and expansion. Although some trigger words are extracted in the second step, the entire trigger words set is firstly, not complete, and secondly, contains some noisy trigger words. The trigger words set is not complete because it only contains verb triggers. For example, there are nominal triggers, such as *marriage*, which are not in the trigger words set. The general idea to filter noisy trigger words and expand non-verb trigger words is to use word embedding to map events in FreeBase to Frames in FrameNet. For example, *appoint.v* is a Lexical Unit of *Appointing* frame in FrameNet. It can be mapped to *people.appointment* events in FreeBase. In this way, the paper[6] finish filtering by removing some verbs and finish expansion by using nouns with high confidence in the mapped frame.

The fourth step is automatically labeled data generation. There are two assumptions at this step. First of all, for an event in a sentence, all key arguments in FreeBase and the corresponding trigger word express the event. Secondly, arguments occur in that sentence are likely to play the corresponding roles in that event. Based on these two assumptions, we can use Soft Distant supervision methods to generate data.

3.2.2 Generate labeled data for event extraction only by identifying key arguments

Traditional approaches of event extraction require trigger word extraction and assign an event type. For example, in Figure 2, trigger word *sold* is extracted and then be assigned with a type of *business.acquisition*. However, this paper[7] argues that identification of trigger is not indispensable to determine the event type. While event triggers are useful, they do not always need to be explicitly captured. Several key arguments together also imply event type. As shown in Figure 2 and 3, the three instance values in the sentence, which are *Remedy Corp*, *BMC Software*, and *2004*, can be mapped to three roles of the event respectively. Not only each instance value describes the role of an argument in the sentence, but also they defines *business.acquisition* event comprehensively.

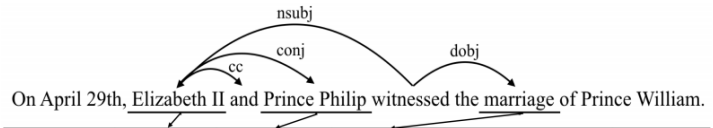
Since this approach only identifies key arguments to generate labeled data, the pipeline of data generation is simpler than the approach discussed in 3.1.1:

- Identify key arguments from a CVT table entry
- Generate labeled data based on existing structured tables or lists.

The first step is to extract key arguments. Similar to the definition of key argument in 3.1.1, the key argument here is defined as an argument that plays an important role in one event, which helps to distinguish with other events. However, this paper[7] uses a different method to determine if an argument is a key argument. The first measurement is **importance score**, to measure importance of arguments. For an argument to an event type, the importance score is defined as following:

$$Icvt, arg = \log(count(cvt, arg)count(cvt)count(arg))$$

The $count(cvt)$ is the number of instances (event mentions) in the CVT table, $count(arg)$ is the number of times that argument appearing in all CVT types with in a CVT table, and $count(cvt, arg)$ is the number of cvt instances that contain that argument across all tables. Secondly, this paper[7] argues that time-related arguments are useful to determine the event type, so it always includes time-related argument in key argument set.



id \ property	spouse	spouse	type_of_union	location_of_ceremony	from
m.028zms8	Elizabeth II	Prince Philip	marriage	Collegiate Church	1947/11/20

Figure 5: Dependency parse tree of a given sentence.

Secondly, the paper[7] remove sentences from the generated dataset in which dependency distance between any two key arguments are greater than 2. The distance between two key arguments is defined as the number of hops in the dependency parse tree. For example, as shown in Figure 5, the distance between argument Elizabeth II and Prince Philip is 1. In general, the strategy of key argument selection can be summarized as following:

- Compute importance score and select top half arguments that have the highest importance scores.
- Always select time-related argument to key argument set.
- Remove sentences from the generated dataset t in which dependency distance between any two key arguments are greater than 2.

The second step is to generate training data. The paper[7] takes in existing structured tables or lists that are organized in a way like FreeBase CVT tables. There are two stages in this step. The first stage is to determine the key arguments for each entry within that type. The general philosophy is to produce a set of rules to be used for data labeling, where each rule contains the event type, key arguments and non-key arguments given by a structured table entry. Furthermore, the paper[7] uses alias information to match two arguments that have different surface names but refer to the same entity. For example, *Microsoft* and *MS* are different surface names but they refer to the same company. The second stage is labeling each individual sentence from the target dataset. The general idea is straightforward: traverse all rules from generated rule set, and check if the target sentence contains all the key arguments specified by a rule.

Although this paper[7] introduces an approach to generate labeled data by identifying less information, it also has some limitations. The first limitation is that the approach relies on structured table or lists rather than raw text data to automatically label text. The second limitation is that the performance of this approach can be improved by introducing pronoun resolution and entity co-reference resolution on target sentences which we will further discuss in following sections.

4. CONTEMPORARY APPROACHES EVALUATION

In this section, we mainly will give introduction and evaluations to some contemporary promising open domain event extraction paradigms. We will give evaluations based on different paradigms and advantages or disadvantages of each models.

4.1 Event Schema Extraction By Trigger Clustering

Event extraction is always a significant task in Text Mining and Information Retrieval. One of the existing approach for event extraction is **ACE**, which extracts event by manually define an event schema based on the needs of potential users. The left part of Figure 6 shows an event schema in ACE. However, such process is very expensive because consumers and experts need to check a lot of data before specifying types of events and argument roles. Furthermore, they need to write annotation guidelines for each type in the schema. The paper[8] introduces a new approach for event extraction as shown in the right part of Figure 6. To extract events from sentence, it firstly extracts trigger words and corresponding arguments in the sentence. For each trigger

word, clusters it and use the cluster name as event type. Then, it assign arguments based on the event types.

The most important task in this approach is the representation of triggers. Basically, trigger words are represented as distributional vectors. Furthermore, there are two guidelines for word embedding. The first one is that event trigger that occurs in similar contexts and shares the same sense tend to have similar types, so distributional vectors of a trigger word should include its own semantic information. The second one is that trigger type is dependent on its arguments, their roles, and other words contextually connected to the trigger, so distributional vectors of a trigger word should also include its argument semantic information. To get argument semantic information, the paper[8] uses semantic relations to specify how the distributional semantics of relevant context words contribute to the overall event structure representation.

Here are the general steps to generate clusters, find out each cluster arguments and name each clusters.

The first step is to identify candidate triggers and arguments. To identify triggers, first consider all noun and verb concepts that are assigned an OntoNotes sense by WSD as candidate event triggers. Remaining concepts that match both verbal and nominal lexical unit in the FrameNet are considered as candidate event triggers. In this way, some nominal triggers like *marriage* and *wedding* can be identified. To identify arguments for each trigger, the paper[8] uses Abstract Meaning Representation (AMR) to identify all arguments.

The second step is to represent trigger and argument in distributional vectors. First apply WSD to link each word to its sense in WordNet. Then map WordNet sense output to OntoNotes senses. For each trigger candidate, map it to OntoNotes sense and learn distinct embedding for each sense. In general, word embeddings are learned from a large dataset by continuous skip-gram model. For arguments, the paper[8] uses general lexical embeddings for arguments.

The third step is event structure composition and representation. For each event trigger, apply a series of compositional functions to generate triggers event structure representation. Each compositional function is specific to a semantic relation, and function is operated over vectors in the embedding space. For each argument, its representation is generated as a by-product.

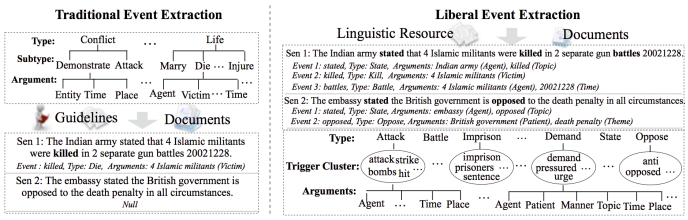


Figure 6: Traditional event extraction and approach introduced by this paper[8].

The fourth step is to use a joint constraint clustering framework to cluster arguments and triggers. There is a hypothesis for triggers that if two triggers arguments have the same type and role, they are more likely to belong to the same type. Thus, the paper[8] uses a constraint function to enforce interdependent triggers and arguments to have coherent types. For clustering algorithm, the paper[8] designs a joint constraint clustering approach, which iteratively produces new clustering results based on constraint, to get cluster result.

The last step is to name each cluster of triggers and each triggers arguments. The paper[8] resolves this problem by operating mapping between meaning representation and semantic role descriptions in KB.

4.2 Deep Learning Approaches

Recent years, many deep learning algorithms, including various neural networks, have also impacted the field of open domain event extraction. Specifically, after the creation and development of some efficient algorithms of word representations in vector space [15], even convolutional neural networks, which previously almost mostly used in computer vision, can be used for open event extraction. These deep learning approaches also have provided many promising solutions to the event extraction. Following we mainly analyze two typical usage of neural networks in the open domain event extraction tasks. Even if these two methods use other methods such as Question Answering, the core for the methods are usage of deep learning techniques.

4.2.1 Recurrent Neural Nets Based Approach

Traditional relation extraction system can extract relations from unstructured corpus given the fact that relations are predefined. Two major approaches to train extraction model are crowdsourcing [1] and distant supervision [2]. But it is incapable for relations that are not predefined. To solve this issue, the paper[9] takes an alternative approach by incorporating latest progresses in the field of reading comprehension and demonstrate that the approach can generalize to defined relation types with high accuracy as well as being able to identify some of the unseen relation types. The main idea introduced by the paper can be divided into following points:

- Pre-conditions (fact showed in the paper): relation extraction can be reduced to answering simple reading comprehension questions, by designing one or more natural-language questions for each argument slots.
- Recent reading comprehension techniques (mainly nerual networks) can be extended to learning relation models.
- For given relation models, distant supervision can be combined with designed crowd-source questions to generate large effective training set.
- Zero-shot learning approach can be applied to handle new relation types that are only specified at test-time, whose labeled training examples are not provided.

Given a relation type $R(x, y)$, where R is a relation and x, y are entities related by relation R. A parametrized natural-language question q_x can be constructed such that once the entity x and the text where x is mentioned are given, y can queried as the answer of q_x or null (which indicates such relation R does not exist). The below figures shows some real examples.

Existing reading-comprehension techniques generally formulate problem with the assumption that an answer is always present in the given text. But this assumption typically

does not hold in the relation extraction context. Thus, it is necessary to extend existing reading comprehension models with the ability to indicate null/empty answers. The authors starts with the original BiDAF recurrent neural network model, which output confidence score for potential start and end positions respectively. By concatenate confidence score vector can be formed for all potential answers of one question and then via Softmax, confidence score vector can be transformed into pseudo probability vectors p_s and p_e, which indicates the probability that the answers start at position s and end at position e. And the probability of the answer is the text that starts at s and ends at e is simply $p_s \odot p_e$ (where $\odot$ represents Hadamard product). The authors then append a bias term b at the end of p_s and p_e to account for answer being null/empty. By this way, the probability null/empty answer can be formulated as $p_s \odot p_e$ where $s == e$. With the bias term, the model become sensitive to the raw confidence scores but enables a per-example decision, a global minimum threshold p_min can then be set to further decide if a question has no answer.

Consider the slot filling challenge in relation extraction, give a relation R, entity e and a context sentence s, the goal is to find either an empty set, or a set of a such that $R(e, a)$ holds for all a in the set. Through read comprehension, a template question q can be constructed by assuming the position of e as a variable x, and q can then be used to query for a and for each e that can be used in the relation R. For example, consider the relation occupation, the entity Steve Jobs, and the sentence Steve Jobs was an American businessman, inventor, and industrial designer. A question q What did x do for a living? can be used for all relevant entity e, such as What did Steve Jobs do for a living?. This process, which is defined as schema querification, is essentially an order of magnitude more efficient doing each entity individually, because all of the instances can be automatically annotated at the same time. But since such formulation will implicitly encourage the annotator to figure out the semantic meaning behind it, an additional verification phase is added by sampling additional sentences and instantiating each question template with the new entity e. Those additional sentences are asked to the annotators to validate the previous schema querficiation results.
In the zero-shot scenario, for a previously unseen relation $R_(n+1)$, it can be first queried into a question q_x, and then instantiate into concrete questions with each possible entity e_x. Each successful answering that resulting in an non-null answer a_x by the trained reading comprehension model, a new relation $R_(n+1)$ (e_x, a_x) is extracted.

4.2.2 Convolutional Neural Nets Based Approach

This section we mainly introduce a paper based on using CNN with Zero-Shot Learning [10]. From the unstructured text data, many traditional supervised methods usually do not have ability to handle new event types since new event types mean cannot use annotations for old events. Specifically, those approaches always make the event extraction as classification problems which encode features merely on the similarity among input event mentions and annotated event mentions. Event type and argument roles in the models play atomic symbol role which enforce limitations. However, in the paper authors propose a new approach which can help to solve problems in this situation.

The interesting thing they found is that both event types and event mentions can be represented by some form of structure. For example, if we represent each event mention with Abstract Meaning Representation structures and event type with Entity Relation Entity structures, we can see that there is association between structures of event mention and event type. Event mention in its event type share similar structure information. Similar to the theory "semantics of an event structure can be generalized and mapped to event mention structures in systematic and predictable way" [22]. Thus, they took an approach mainly by mapping each mention to its semantically closest event type in ontology.

The implementation idea is the Zero-Shot Learning. Zero-Shot Learning is a technique that can solve a classification problem when there is not enough training labels available for some labels which can fit into the situation of event extraction. In computer vision, usually when using the Zero-Shot Learning, images and labels are mapped into a multi-dimensional vector space separately and then learn a regression model to predict unseen labels for given images. Thus, for event extraction, they treat event types with annotated event mentions as seen types, others as unseen types, utilizing neural network architecture jointly learns and maps the structural representations of both into shared semantic space by the minimization of the distance between event types and corresponding event mentions. If an event mention is unseen, its structure will be first projected into the semantic space and then assign event types with the top-ranked similarity values.

For this approach, there are mainly two advantages: The first one is the transferring knowledge from existing seen types to unseen types without additional annotation effort. The second one is that some existing ontologies provide a wide range of event types which also extend the scope of extraction by using this method. Following are the general pipeline of the approach:

- Given a sentence s, based on AMR parsing (As the Table 2 shows), first identify triggers and arguments and build a structure S_t. Each one has set of tuples, for example (dispatch-01, :ARG0, China). For each AMR relation we can use a matrix to represent it.
- For each event type y, also construct type structure S_y, composing each type and argument into a tensor with (Transport-Person, Destination) etc.
- Using a shared CNN to generate event mention representation V_{S_t} and event type representation V_{S_y} by minimizing the semantic distance among the corresponding event type and event mention. Thus, each mention should be closest to its annotated type after the jointly mapping.
- After the training, incoming new event mention we can project it into semantic space which can find its closest event type.

The identification of candidate triggers and arguments are based on AMR Parsing and apply word sense disambiguation tool to distinguish the word sense and link each sense to OntoNotes.

For each candidate t and type y, based on AMR parsing and defined roles and take the types as root, constructing the structure S_t and S_y. Each structure are composed of a

Table 2: event-related AMR Relations

Categories	Relations
Core roles	ARG0, ARG1, ARG2, ARG3, ARG4
None-core roles	mod, location, instrument, poss, manner...
Temporal	year, duration, decade, weekday, time
Spatial	destination, path, location

set of tuples. Following are two approaches to incorporate the semantics of relations into the two words of each tuple. Following are the details on how to represent two structures in vector space:

- Event Mention Structure. For each tuple $u = (w_1, \lambda, w_2)$ in an event mention structure, using matrix to represent each AMR relation and compose the semantics of AMR relation λ to the w_1 and w_2 as: $V_u = [V'_{w_1}; V'_{w_2}] = f([V_{w_1}; V_{w_2}] \cdot M_\lambda)$ where $V_{w_1}, V_{w_2} \in \mathbf{R}^d$ which are the vector representations of the words. $[;]$ denotes the concatenation of two word vectors. $M_\lambda \in \mathbf{R}^{2d \times 2d}$ is the matrix representation of AMR relation λ. V_u is the representation of tuple u [10].
- Event Type Structure. For each tuple $u' = (y, r)$ in an event type structure where y, r denote event type and argument role respectively. Thus, using a single tensor to represent the implicit relation, $V_{u'} = [V'_y; V'_r] = f([V_y; V_r] \cdot U^{[1:2d]} \cdot [V_y; V_r])$ where V_y, V_r are corresponding word vectors. $U^{[1:2d]} \in \mathbf{R}^{2d \times 2d \times 2d}$ is a 3-order tensor which incorporates the implicit relation [10].

After we can represent the structures in vectors, we can create appropriate cost function which then can be used in training a shared weight convolutional neural network and thus joint event mention and type label embedding.

Mapping each argument of event mention to specific role is similar to previous joint event mention and type label embedding. For example, China $\Rightarrow$ Agent. Given a trigger word t and candidate argument a, we first get $S_a = (u_1, u_2...)$ which is a path connects t and a and consists p tuples [10]. Similar to last part, we can incorporate roles and the trigger types together and train a similar shared weight CNN.

Given a new event mention t', computing its mention structure first as $S_{t'}$ and all event type structure representations $S_Y = \{S_{y1}, S_{y2}...\}$ from what have trained. Ranking all event types based on the similarity scores with mention t'. $\hat{y}(t', 1) = \arg\max_{y \in Y} cos([V_{t'}; V_{S_{t'}}], [V_y; V_{S_y}])$ and thus we can assign the appropriate event type to the mentions.

After determining the type, for each candidate argument, adopt the same ranking function to find the most appropriate role from the role set defined for the type.

This paper proposes an useful and relatively novel pipeline of approaches for open domain event extraction [10]. First it is possible to use contemporary methods to transfer the relation and type structures into high quality lower space vectors. Second we can use these transfered structures in zero-shot learning which can give better result even if lack of training examples. Third, different structures in event extraction can be trained together with a shared weight CNN. These ideas would be helpful in the current approaches of utilization zero-shot and neural networks in open domain event extraction. One of the main disadvantages of deep learning approach is that it requires relatively large amount of training data. Thus, if one can augment the training data by utilizing the techniques we mentioned in section 3.2, then do the training; the result would might be bettered.

4.3 Zero-Shot Learning

Two papers introduced in section 4.2 both use the Zero-Shot Learning. This is a relatively new technique which can solve a classification problem when there is not enough training data available for some labels. Since Zero-Shot Learning usually is implemented by deep learning techniques, we put main part of Zero-Shot Learning in section 4.2.

4.4 Schema Querficiation

In the paper[9] mentioned in section 4.2.1, the crowdsourcing technique schema querfication is used to transforming pre-existing relation-extraction datasets into a reading-comprehension datasets, and also providing more efficient annotating through creating templated questions. This helps us to effectively perform open domain event extraction via the latest reading comprehension framework. More detail is discussed in section 4.2.1.

5. WIKIFICATION

Wikification, commonly referred to as Disambiguation to Wikipedia (D2W), is the process of identifying concepts and entities in text and disambiguating them into the corresponding Wikipedia title pages. Because of the comprehensiveness nature of Wikipedia, wikification is one of the mostly used methods for entity linking, and thus we survey in this area hoping it can be used in open domain event extraction. In this section, we study a well-known paper[19] in wikification, which briefly summarized and improved on traditional approaches in D2W.

Disambiguating entities in a given context has always been a fundamental problem in natural language processing. Wikipedia, as increasingly comprehensive knowledge base, has became a popular target for disambiguation (usually referred as wikification) due to the fact that its link structure can also served as information during the disambiguation process. Major traditional approaches mainly fall into two categories: local approach or global approach.

5.1 Local Approach

In the local approach, each mention is disambiguated individually, utilizing common features such as textual similarity between the given document and each candidates Wikipedia disambiguation page. A scoring function phi is designed to reflecting the likelihood of a candidate title (t_j) is the correct disambiguation of each mention (m_i). And then trying to solve it as a optimization problem that maximizing such likelihoods. Classic works involve kernel SVM, lexical overlapping feature and Naive Bayes classifier on hyper-link information as ground truth. (t_j and m_i changed)

5.2 Global Approach

The global approach is derived based on the assumption such that, if all mentions inside the same document are disambiguated correctly, those disambiguations will tend to form a set of related concepts with certain level of coherence. Thus

an additional term is added to the local approaches scoring function to account for the global coherence. But with the new term, original optimization problem will become NP-hard and an approximation disambiguation context is usually generated and then define a pairwise psi term to account for semantic relatedness in the generated context to simplify the problem. The common approach for generate such approximation is utilizing pairwise relatedness obtained from Wikipedia link graph. There are two well studied approaches in generating disambiguation context and defining the semantic relatedness goes separate ways. On the one hand, Milne and Witten[20] defines the approximate disambiguation context by taking the set of all unambiguous surface name in the input text, and title relatedness is computed as NGD (Normalized Google Distance). This approach has its limitation because it relies on the presence of unambiguous mentions in the input document/text, which is usually not a comprehensive assumption. On the other hand, Cucerzan[21] uses all plausible disambiguations of named entities in the text and formulate title relatedness based on overlap in categories and incoming links. But this approach also brings an disadvantage because it inevitably includes irrelevant titles into the approximate disambiguation context, creating noise.

5.3 GLOW

The authors proposed a two-stage optimization framework that utilizes both local and global features. During the first stage, a ranker is trained to generate top non-null disambiguation candidates for a entity mention. And then a linker is used in the second stage, to determine which candidate title should this entity mention disambiguated to, or null. The null possibility is handled specifically to account for one of the following three cases: 1) the queried mention does not have a corresponding Wikipedia page, 2) the queried mention does have a corresponding Wikipedia page, but is not included in the candidate list and 3) the ranker made a completely wrong decision that chose an incorrect disambiguation. Both linker and ranker is trained as linear SVM.

5.4 Ranker

Wikipedia hyper-links is used and the author further utilize the anchor-title index which is computed from crawler-ed Wikipedia. Each distinct hyper-link anchor text is mapped to its target Wikipedia titles. To improve the efficiency, a shallow parser and named entity recognition system is used to prune the substring search space, by considering only named entities, noun-phrase chunks and all sub-expression of up to 5 tokens of the noun-phrase chunks. Then by querying the anchor-text index, top 20 most frequent target pages is selected based on the proper trade-off between accuracy and efficiency. The ranker is trained based on both local and global features described as follow:

5.5 Generalization of Local and Global Features

The local and global features used in GLOW is summarized below, the details is listed in figure 7.

- Baseline feature (during ranker phrase): $P(t|m)$, the fraction of times the title t is the target page for an anchor text m and $P(t)$, gives the fraction of all Wikipedia articles that link to t.
- Text based local features (account for textual similarity): Text(t), top-200 token TF-IDF summary of the context within which title t was hyper-linked into Wikipedia (denoted by Context(t)). Text(d), TF-IDF representation of a 100-token window around m in the input document. Weighted version of above features to reweight TF-IDF vectors.
- NGD (Normalized Google Distance)
- PMI (Pointwise Mutual Information)
- Average and maximum of above (in longer document, which may cover more sub-topics, maximum might be more informative than simple average)

Baseline Feature: $P(t\|m)$, $P(t)$
Local Features: $\phi_i(t, m)$
cosine-sim(Text(t),Text(m)) : Naive/Reweighted
cosine-sim(Text(t),Context(m)): Naive/Reweighted
cosine-sim(Context(t),Text(m)): Naive/Reweighted
cosine-sim(Context(t),Context(m)): Naive/Reweighted
Global Features: $\psi_i(t_i, t_j)$
$I_{[t_i-t_j]}$*PMI(InLinks(t_i),InLinks(t_j)) : avg/max*
$I_{[t_i-t_j]}$*NGD(InLinks(t_i),InLinks(t_j)) : avg/max*
$I_{[t_i-t_j]}$*PMI(OutLinks(t_i),OutLinks(t_j)) : avg/max*
$I_{[t_i-t_j]}$*NGD(OutLinks(t_i),OutLinks(t_j)) : avg/max*
$I_{[t_i \leftrightarrow t_j]}$ *: avg/max*
$I_{[t_i \leftrightarrow t_j]}$*PMI(InLinks($t_i$),InLinks($t_j$)) : avg/max*
$I_{[t_i \leftrightarrow t_j]}$*NGD(InLinks($t_i$),InLinks($t_j$)) : avg/max*
$I_{[t_i \leftrightarrow t_j]}$*PMI(OutLinks($t_i$),OutLinks($t_j$)) : avg/max*
$I_{[t_i \leftrightarrow t_j]}$*NGD(OutLinks($t_i$),OutLinks($t_j$)) : avg/max*

Figure 7: GLOW Features

5.6 Linker

The linker adopts the same set of features from ranker with additional features that are relevant to linkers job:

- Confidence of the ranker in t with the second-best disambiguation t, to account for the case that ranker might made a mistake.
- Entropy of the distribution of $P(t|m)$.
- The percent of Wikipedia titles in which m appears hyper-linked versus the percent of times m appears as plain text.
- Whether m was detected by NER as a named entity.
- Good-Turing estimate of how likely m is to be out-of-Wikipedia concept based on counts in $P(t|m)$.

6. COREFERENCE

Coreference resolution is the task of recognizing which mentions in text refer to the same real-world entity, and it is indispensable in the heterogeneous network construction. For example:

Tom is eating an apple, but he likes pear the best.

If we directly use this sentence to extract event and construct a graph, we will lost information that *Tom likes pear*

the best, because machine would treat *Tom* and *he* as different entities. With coreference resolution, machine will know that the word *he* is referred to *Tom* and capture the information *Tom likes pear the best*. One approach of mainstream coreference system is agglomerative clustering. Firstly, treat each mention as a singleton cluster and then repeatedly merge clusters of mentions that referred to the same entity. Such approach captures entity-level information, in other words, it captures features between cluster rather than mentions. This paper[16] mainly introduces a new approach based on deep neural network and agglomerative clustering. It uses learned continuous features instead of a smaller number of categorical ones to cluster.

The general idea of the paper[16] is to use a single neural network that learns which coreference cluster merges are desirable. The network is composed with several neural networks, which are mention-pair encoder, cluster-pair encoder, cluster-ranking model. Mention-pair encoder is used to produce distributed representations for pairs of mentions by feeding the neural network with relevant features. Cluster-pair encoder is used to produce distributed representations of clusters by pooling over the representations of relevant mention pairs. Cluster-ranking model scores pairs of cluster by passing their representations through a single neural network layer. Other than these components, this paper[16] also trained a mention-ranking model. Although it is not a component of system structure, its parameters are used to initialize the cluster ranking model. Furthermore, the paper[16] use scores to prune mentions in candidate cluster when doing clustering to increase speed.

The input of encoder is an embedding of mention m, ms features, an embedding of candidate antecedent *a*, *as* features, and document and pair features. The candidate antecedent can be any mention that occurs before m in the document or NA (no antecedent for the mention). For mention and antecedent embedding, each group of words are represented as an average vector of word in that group, and each word is represented as a 50-dimensional vectors produced by word2vec. For each mention and pair of mentions, embedding features (head word, dependency parent, etc.), additional mention features (type, position, etc.), document genre (broadcast news, web data, etc.), distance features (distance between mentions), speaker features (if mentions have the same speaker and string matching features are also represented as distributional vectors. The input gets passed through three hidden layers of ReLU units and output the mention-pair.

The cluster-pair encoder generates the distributed representation of two cluster of mentions. It combines the information contained in the matrix of mention-pair representation in each cluster, which is done by pooling operation.

The mention-ranking model learns effective weights for the mention-pair encoder, so that it can be served as pretraining for cluster ranking model. Its scores can be used to determine if a clustering decision is clearly right or wrong to decrease the search space of cluster ranking model. The model takes a mention m and a candidate antecedent a and apply a single fully connected layer to get a score that represents their compatibility for coreference.

The disadvantage of mention-ranking model is that it only considers local information between pairs of mentions, so it cannot capture entity-level information. Thus, the paper[16] introduces cluster ranking model to address the problem.

- Cluster ranking policy network: The cluster ranker iterates through every mention in the document and merge the current mentions cluster with a preceding one or perform no action. At a start state, each cluster contains a single mention, and the cluster ranker make decisions to merge of pass. Such action decision can be learned as probability distributions.

- Easy first cluster ranking: The paper[16] sorts the mentions in descending order by their highest scoring candidate coreference link according to mention-ranking model. In this way, easier decisions are made earlier than harder decisions. Such approach improves the performance of the coreference resolution system. Furthermore, the paper[16] prunes the set of candidate antecedences for each mention m. In other words, only high-scoring candidate antecedents are considered. Such methods increase the speed of learning and inference.

7. CONCLUSIONS

In this survey we give introduction for open domain event extraction and evaluations on recent promising approaches. Open domain event extraction techniques can significantly better the result of information extraction. It can help mine structured text knowledge specifically during contemporary text data explosion period. According to our summarization of open domain event extraction, some knowledge bases such as Wikipedia will be helpful in the open domain event extraction situation. Existing ontologies can better the mining result. Apart from the existing knowledge bases and ontologies, training data for open domain extraction is also crucial, specifically for deep learning methods which usually require relatively larger amount of data for training. In our survey, those two methods to automatically generate open domain event extraction training data with relatively low expense and high accuracy.

Although some existing knowledge base and ontologies, and automatically generated training data are helpful, some other natural language processing techniques also play much more important roles. The entity linking and co-reference resolution, for instance, if used properly, they can help construct quality and dense information network from text corpus and thus meets the requirements of performing an open domain event extraction.

In the survey, we also introduced some current promising approaches on open domain event extraction including deep learning, Zero-Shot Learning, event schema extraction by trigger clustering and schema querification. These approaches might in the future take more important roles in open domain event extraction.

8. REFERENCE

1 Hogenboom, Frederik, et al. "An overview of event extraction from text." Workshop on Detection, Representation, and Exploitation of Events in the Semantic Web (DeRiVE 2011) at Tenth International Semantic Web Conference (ISWC 2011). Vol. 779. 2011.

2 McClosky, David, Mihai Surdeanu, and Christopher D. Manning. "Event extraction as dependency parsing."

Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1. Association for Computational Linguistics, 2011.

3 Cohen, K. Bretonnel, et al. "High-precision biological event extraction with a concept recognizer." Proceedings of the Workshop on Current Trends in Biomedical Natural Language Processing: Shared Task. Association for Computational Linguistics, 2009.

4 Ritter, Alan, Oren Etzioni, and Sam Clark. "Open domain event extraction from twitter." Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2012.

5 Okamoto, Masayuki, and Masaaki Kikuchi. "Discovering volatile events in your neighborhood: Local-area topic extraction from blog entries." Asia Information Retrieval Symposium. Springer, Berlin, Heidelberg, 2009.

6 Chen, Yubo, et al. "Automatically labeled data generation for large scale event extraction." Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Vol. 1. 2017.

7 Zeng, Ying, et al. "Scale Up Event Extraction Learning via Automatic Training Data Generation." arXiv preprint arXiv:1712.03665 (2017).

8 Huang, Lifu, et al. "Liberal event extraction and event schema induction." Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Vol. 1. 2016.

9 Levy, Omer, et al. "Zero-shot relation extraction via reading comprehension." arXiv preprint arXiv:1706.04115 (2017).

10 Huang, Lifu, et al. "Zero-Shot Transfer Learning for Event Extraction." arXiv preprint arXiv:1707.01066 (2017).

11 Chen, Danqi, et al. "Reading wikipedia to answer open-domain questions." arXiv preprint arXiv:1704.00051 (2017).

12 Watanabe, Yusuke, Bhuwan Dhingra, and Ruslan Salakhutdinov. "Question Answering from Unstructured Text by Retrieval and Comprehension." arXiv preprint arXiv: 1703.08885 (2017).

13 Hunter, Lawrence, et al. "OpenDMAP: an open source, ontology-driven concept analysis engine, with applications to capturing knowledge regarding protein transport, protein interactions and cell-type-specific gene expression." BMC bioinformatics 9.1 (2008): 78.

14 Hachey, Ben, et al. "Evaluating entity linking with Wikipedia." Artificial intelligence 194 (2013): 130-150.

15 Mikolov, Tomas, et al. "Efficient estimation of word representations in vector space." arXiv preprint arXiv: 1301.3781 (2013).

16 Clark, Kevin, and Christopher D. Manning. "Improving coreference resolution by learning entity-level distributed representations." arXiv preprint arXiv:1606.0 1323 (2016).

17 Liu, Angli, et al. "Effective crowd annotation for relation extraction." Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. 2016.

18 Hoffmann, Raphael, et al. "Knowledge-based weak supervision for information extraction of overlapping relations." Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1. Association for Computational Linguistics, 2011.

19 Ratinov, Lev, et al. "Local and global algorithms for disambiguation to wikipedia." Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1. Association for Computational Linguistics, 2011.

20 Witten, Ian H., and David N. Milne. "An effective, low-cost measure of semantic relatedness obtained from Wikipedia links." (2008): 25-30.

21 Cucerzan, Silviu. "Large-scale named entity disambiguation based on Wikipedia data." Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL). 2007.

22 Pustejovsky, James. "The syntax of event structure." Cognition 41.1-3 (1991): 47-81.

23 Baker, Collin F., Charles J. Fillmore, and John B. Lowe. "The berkeley framenet project." Proceedings of the 17th international conference on Computational linguistics-Volume 1. Association for Computational Linguistics, 1998.

24 Liu, Shulin, et al. "Leveraging framenet to improve automatic event detection." Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Vol. 1. 2016.