Liberal Event Extraction and Event Schema Induction

Lifu Huang[1], **Taylor Cassidy**[2], **Xiaocheng Feng**[3],
Heng Ji[1], **Clare R. Voss**[2], **Jiawei Han**[4], **Avirup Sil**[5]
[1] Rensselaer Polytechnic Institute, [2] US Army Research Lab,
[3] Harbin Institute of Technology, [4] Univerisity of Illinois at Urbana-Champaign,
[5] IBM T.J. Watson Research Center
[1]{huangl7,jih}@rpi.edu,
[2]{taylor.cassidy.civ,clare.r.voss.civ}@rpi.edu,
[3]xcfeng@ir.hit.edu.cn, [4]hanj@illinois.edu, [5]avi@us.ibm.com

Abstract

We propose a brand new "Liberal" Event Extraction paradigm to extract events and discover event schemas from any input corpus simultaneously. We incorporate symbolic (e.g., Abstract Meaning Representation) and distributional semantics to detect and represent event structures and adopt a joint typing framework to simultaneously extract event types and argument roles and discover an event schema. Experiments on general and specific domains demonstrate that this framework can construct high-quality schemas with many event and argument role types, covering a high proportion of event types and argument roles in manually defined schemas. We show that extraction performance using discovered schemas is comparable to supervised models trained from a large amount of data labeled according to predefined event types. The extraction quality of new event types is also promising.

1 Introduction

Event extraction aims at identifying and typing trigger words and participants (arguments). It remains a challenging and costly task. The first question is what to extract? The TIPSTER (Onyshkevych et al., 1993), MUC (Grishman and Sundheim, 1996), CoNLL (Tjong et al., 2003; Pradhan et al., 2011), ACE [1] and TAC-KBP (Ji and Grishman, 2011) programs found that it was feasible to manually define an event schema based on the needs of potential users. An ACE event schema example is shown in Figure 1. This process is very expensive because consumers and expert linguists need to examine a lot of data before specifying the types of events and argument roles and writing detailed annotation guidelines for each type in the schema. Manually-defined event schemas often provide low coverage and fail to generalize to new domains. For example, none of the aforementioned programs include "*donation*" and "*evacuation*" in their schema in spite of their potential relevance to users.

In this paper we propose *Liberal Event Extraction*, a new paradigm to take humans out of the loop and enable systems to extract events in a more liberal fashion. It automatically discovers a complete event schema, customized for a specific input corpus. Figure 1 compares the ACE event extraction paradigm and our proposed Liberal event extraction paradigm.

We use the following examples to explain and motivate our approach, where event triggers are in bold and arguments are in italics and underlined:

E1. *Two Soldiers* were **killed** and *one* **injured** in the close-quarters **fighting** in *Kut*.

E2. *Bill Bennet*'s glam *gambling* **loss** changed my opinion.

E3. Gen. Vincent Brooks announced the **capture** of *Barzan Ibrahim Hasan al-Tikriti*, telling reporters he was an adviser to Saddam.

E4. This was the *Italian ship* that was **captured** by *Palestinian* terrorists back in 1985.

E5. *Ayman Sabawi Ibrahim* was **arrested** in *Tikrit* and was **sentenced** to life in *prison*.

We seek to cluster the event triggers and event arguments so that each cluster represents a type. We rely on distributional similarity for our clustering distance metric. The distributional hypothesis (Harris, 1954) states that words often occurring in similar contexts tend to have similar meanings. We formulate the following distributional

[1] http://www.itl.nist.gov/iad/mig/tests/ace/

Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, pages 258–268,
Berlin, Germany, August 7-12, 2016.

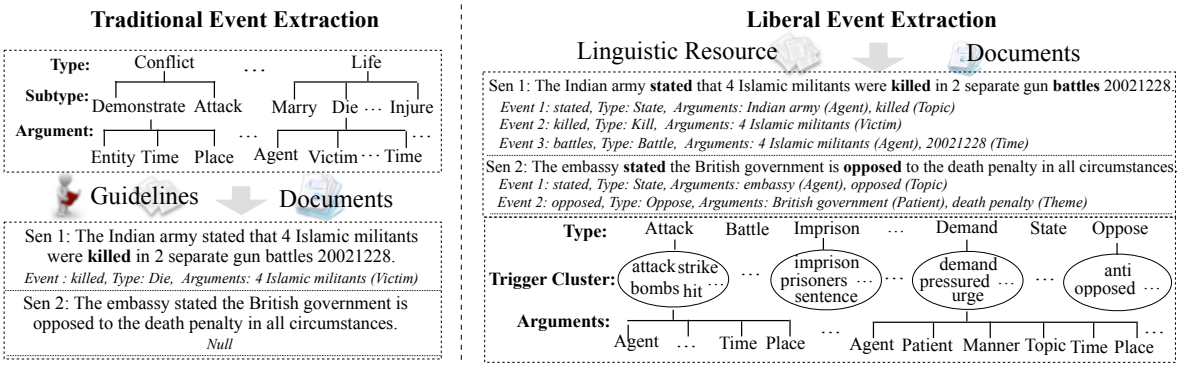

Figure 1: Comparison between ACE Event Extraction and Liberal Event Extraction.

hypotheses specifically for event extraction, and develop our approach accordingly.

Hypothesis 1: *Event triggers that occur in similar contexts and share the same sense tend to have similar types.*

Following the distributional hypothesis, when we simply learn general word embeddings from a large corpus for each word, we obtain similar words like those shown in Table 1. We can see similar words, such as those centered around "*injure*" and "*fight*", are converging to similar types. However, for words with multiple senses such as "*fire*" (shooting or employment termination), similar words may indicate multiple event types. Thus, we propose to apply Word Sense Disambiguation (WSD) and learn a distinct embedding for each sense (Section 2.3).

injure	Score	fight	Score	fire	Score
injures	0.602	fighting	0.792	fires	0.686
hurt	0.593	fights	0.762	aim	0.683
harm	0.592	battle	0.702	enemy	0.601
maim	0.571	fought	0.636	grenades	0.597
injuring	0.561	Fight	0.610	bombs	0.585
endanger	0.543	battles	0.590	blast	0.566
dislocate	0.529	Fighting	0.588	burning	0.562
kill	0.527	bout	0.570	smoke	0.558

Table 1: Top-8 Most Similar Words (in 3 Clusters)

Hypothesis 2: *Beyond the lexical semantics of a particular event trigger, its type is also dependent on its arguments and their roles, as well as other words contextually connected to the trigger.*

For example, in *E4*, the fact that the patient role is a vehicle ("*Italian ship*"), and not a person (as in *E3* and *E5*), suggests that the event trigger "*captured*" has type "*Transfer-Ownership*" as opposed to "*Arrest*". In *E2*, we know the "*loss*" event occurs in a *gambling* scenario, so we can determine its type as loss of money, not loss of life.

We therefore propose to enrich each trigger's representation by incorporating the distributional representations of various words in the trigger's context. Not all context words are relevant to event trigger type prediction, while those that are vary in their predictive value. We propose to use semantic relations, derived from a meaning representation for the text, to carefully select arguments and other words in an event trigger's context. These words are then incorporated into a "global" event structure for a trigger mention. We rely on semantic relations to (1) specify how the distributional semantics of relevant context words contribute to the overall event structure representation; (2) determine the order in which distributional semantics of relevant context words are incorporated into the event structure (Section 2.4).

2 Approach

2.1 Overview

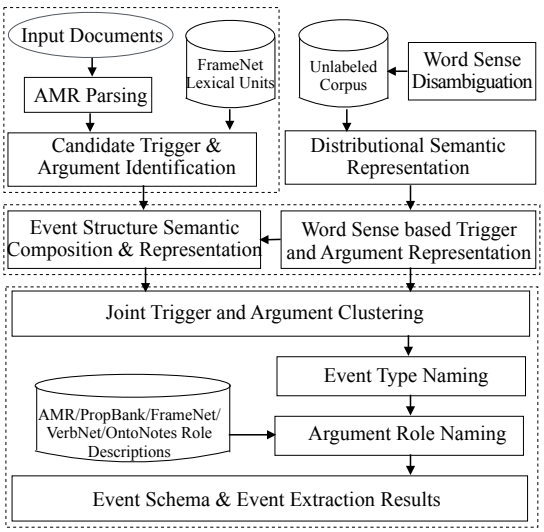

Figure 2: Liberal Event Extraction Overview.

Figure 2 illustrates the overall framework of

Liberal Event Extraction. Given a set of input documents, we first extract semantic relations, apply WSD and learn word sense embeddings. Next, we identify candidate triggers and arguments.

For each event trigger, we apply a series of compositional functions to generate that trigger's event structure representation. Each function is specific to a semantic relation, and operates over vectors in the embedding space. Argument representations are generated as a by-product.

Trigger and argument representations are then passed to a joint constraint clustering framework. Finally, we name each cluster of triggers, and name each trigger's arguments using mappings between the meaning representation and semantic role descriptions in FrameNet, VerbNet (Kipper et al., 2008) and Propbank (Palmer et al., 2005).

We compare settings in which semantic relations connecting triggers to context words are derived from three meaning representations: Abstract Meaning Representation (AMR) (Banarescu et al., 2013), Stanford Typed Dependencies (Marie-Catherine et al., 2006), and FrameNet (Baker and Sato, 2003). We derive semantic relations automatically for these three representations using CAMR (Wang et al., 2015a), Stanford's dependency parser (Manning, 2003), and SEMAFOR (Das et al., 2014), respectively.

2.2 Candidate Trigger and Argument Identification

Given a sentence, we consider all noun and verb concepts that are assigned an OntoNotes (Hovy et al., 2006) sense by WSD as candidate event triggers. Any remaining concepts that match both a verbal and a nominal lexical unit in the FrameNet corpus are considered candidate event triggers as well. This mainly helps to identify more nominal triggers like *"pickpocket"* and *"sin"*.[2]

For each candidate event trigger, we consider as candidate arguments all concepts for which one of a manually-selected set of semantic relations holds between it and the event trigger. For the setting in which AMR serves as our meaning representation, we selected a subset of all AMR relations that specify event arguments, as shown in Table 2. Note that some AMR relations generally do not specify event arguments, e.g. "mode", which can indicate sentence illocutionary force, or "snt" which is used to combine multiple sentences into one AMR graph.[3] When FrameNet is the meaning representation we allow all frame relations to identify arguments. For dependencies, we manually mapped dependency relations to AMR relations and use Table 2.

Categories	Relations
Core roles	ARG0, ARG1, ARG2, ARG3, ARG4
Non-core roles	mod, location, poss, manner, topic, medium, instrument, duration, prep-X
Temporal	year, duration, decade, weekday, time
Spatial	destination, path, location

Table 2: Event-Related AMR Relations.

In E1, for example, "*killed*", "*injured*" and "*fighting*" are identified as candidate triggers, and three concept sets are identified as candidate arguments using AMR relations: "{*Two Soldiers, very large missile*}", "{*one, Kut*}" and "{*Two Soldiers, Kut*}", as shown in Figure 3.

2.3 Trigger Sense and Argument Representation

Based on Hypothesis 1, we learn sense-based embeddings from a large data set, using the Continuous Skip-gram model (Mikolov et al., 2013). Specifically, we first apply WSD to link each word to its sense in WordNet using a state-of-the-art tool (Zhong and Ng, 2010), and map WordNet sense output to OntoNotes senses. [4] We map each trigger candidate to its OntoNotes sense and learn a distinct embedding for each sense. We use general lexical embeddings for arguments.

2.4 Event Structure Composition and Representation

Based on Hypothesis 2, we aim to exploit linguistic knowledge to incorporate inter-dependencies between event and argument role types into our event structure representation. Many meaning representations could provide such information to some degree. We illustrate our method for building event structures using semantic relations from meaning representations using AMR. In Section 3.4 we compare results using Stanford Typed Dependencies and FrameNet in place of AMR.

Let's take *E2* as an example. Based on AMR annotation and Table 2, we extract semantically re-

[2]For consistency, we use the same trigger identification procedure regardless of which meaning representation is used to derive semantic relations.

[3]For relation details, see https://github.com/amrisi/amr-guidelines/blob/master/amr.md

[4]WordNet-OntoNotes mapping from https://catalog.ldc.upenn.edu/LDC2011T03

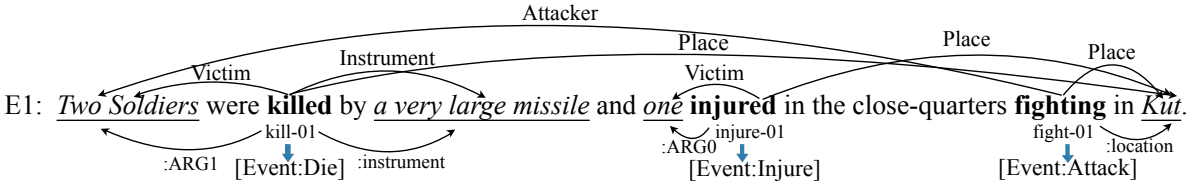

Figure 3: Event Trigger and Argument Annotations and AMR Parsing Results of *E1*.

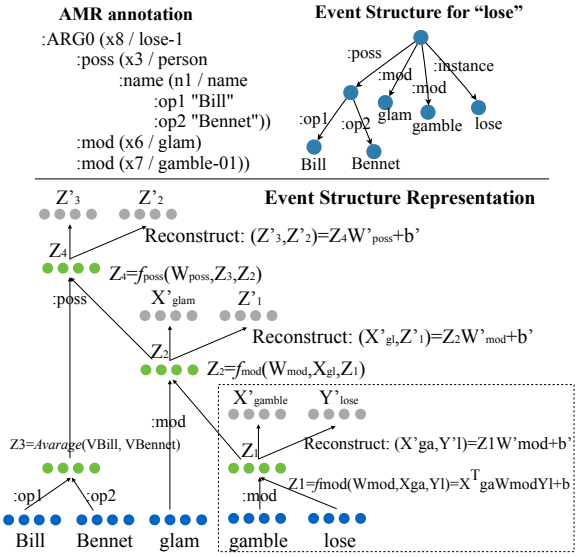

Figure 4: Partial AMR and Event Structure for *E2*.

lated words for the event trigger with sense "*lose-1*" and construct the event structure for the whole event, as shown in Figure 4.

We design a Tensor based Recursive Auto-Encoder (TRAE) (Socher et al., 2011) framework to utilize a tensor based composition function for each of a subset of the AMR semantic relations and compose the event structure representation based on multiple functional applications. This subset was manually selected by the authors as the set of relations that link a trigger to concepts that help to determine its type. Similarly, we selected a subset of dependency and FrameNet relations using the same criteria for experiments using those meaning representations.

Figure 4 shows an instance of a TRAE applied to an event structure to generate its representation. For each semantic relation type r, such as "*:mod*", we define the output of a tensor product Z via the following vectorized notation:

$$Z = f_{mod}(X, Y, W_r^{[1:d]}, b) = [X;Y]^T W_r^{[1:d]} [X;Y] + b$$

where $W_{mod} \in \mathbb{R}^{2d \cdot 2d \cdot d}$ is a 3-order tensor, and $X, Y \in \mathbb{R}^d$ are two input word vectors. $b \in \mathbb{R}^d$ is the bias term. $[X;Y]$ denotes the concatenation of two vectors X and Y. Each slice of the tensor acts as a coefficient matrix for one entry Z_i in Z:

$$Z_i = f_{mod}(X, Y, W_r^{[i]}, b) = [X;Y]^T W_r^{[i]} [X;Y] + b_i$$

We use the statistical mean to compose the words connected by ":op" relations (e.g. "*Bill*" and "*Bennet*" in Figure 4).

After composing the vectors of X and Y, we apply an element-wise sigmoid activation function to the composed vector and generate the hidden layer representations Z. One way to optimize Z is to try to reconstruct the vectors X and Y by generating X' and Y' from Z, and minimizing the reconstruction errors between the input $V_I = [X, Y]$ and output layers $V_O = [X', Y']$. The error is computed based on Euclidean distance function:

$$E(V_I, V_O) = \frac{1}{2} ||V_I - V_O||^2$$

For each pair of words X and Y, the reconstruction error back-propagates from its output layer to input layer through parameters $\Theta_r = (W_r', b_r', W_r, b_r)$. Let δ_O be the residual error of the output layer, and δ_H be the error of the hidden layer:

$$\delta_O = -(V_I - V_O) \cdot f'_{\text{sigmoid}}(V_H^O)$$

$$\delta_H = (\sum_{k=1}^{d} \delta_O^k \cdot (W_r'^k + (W_r'^k)^T) \cdot V_H^O) \cdot f'_{\text{sigmoid}}(V_H^I)$$

where V_H^I and V_H^O denote the input and output of the hidden layer, and $V_H^O = Z$. $W_r'^k$ is the k^{th} slice of tensor W_r'.

To minimize the reconstruction errors, we utilize gradient descent to iteratively update parameters Θ_r:

$$\frac{\partial E(\Theta_r)}{\partial W_r'^k} = \delta_O^k \cdot (V_H^O)^T \cdot V_H^O$$

$$\frac{\partial E(\Theta_r)}{\partial b_r'} = -(V_I - V_O) \cdot f'_{\text{sigmoid}}(V_H^O)$$

$$\frac{\partial E(\Theta_r)}{\partial W_r^k} = \delta_H^k \cdot (V_I)^T \cdot V_I$$

$$\frac{\partial E(\Theta_r)}{\partial b_r} = (\sum_{k=1}^{d} \delta_O^k \cdot (W_r'^k + (W_r'^k)^T) \cdot V_H^O) \cdot f'_{\text{sigmoid}}(V_H^I)$$

After computing the composition vector of Z_1 based on X and Y, for the next layer, it composes Z_1 and another new word vector such as

X_{gl}. For each type of relation r, we randomly sample 2,000 pairs to train optimized parameters Θ_r. For each event structure tree, we iteratively repeat the same steps for each layer. For multiple arguments at each layer, we compose them in the order of their distance to the trigger: the closest argument is composed first.

2.5 Joint Trigger and Argument Clustering

Based on the representation vectors generated above, we compute the similarity between each pair of triggers and arguments, and cluster them into types. Recall that a trigger's arguments are identified as in section 2.2. We observe that, for two triggers t_1 and t_2, if their arguments have the same type and role, then they are more likely to belong to the same type, and vice versa. Therefore we introduce a constraint function f, to enforce inter-dependent triggers and arguments to have coherent types:

$$f(\mathcal{P}_1, \mathcal{P}_2) = \log(1 + \frac{|\mathcal{L}_1 \cap \mathcal{L}_2|}{|\mathcal{L}_1 \cup \mathcal{L}_2|})$$

where $\mathcal{P}_1$ and $\mathcal{P}_2$ are triggers. Elements of $\mathcal{L}_i$ are pairs of the form $(r, \text{id}(a))$, where $\text{id}(a)$ is the cluster ID for argument a that stands in relation r to $\mathcal{P}_i$. For example, let $\mathcal{P}_1$ and $\mathcal{P}_2$ be triggers "*capture*" and "*arrested*" (c.f. Figure 5). If *Barzan Ibrahim Hasan al-Tikriti* and *Ayman Sabawi Ibrahim* share the same cluster ID, the pair $(arg1, \text{id(Barzan Ibrahim Hasan al-Tikriti)})$ will be a member of $\mathcal{L}_1 \cap \mathcal{L}_2$. This argument overlap is evidence that "*capture*" and "*arrested*" have the same type. We define f where $\mathcal{P}_i$ are arguments, and elements $\mathcal{L}_i$ are defined analogously to above.

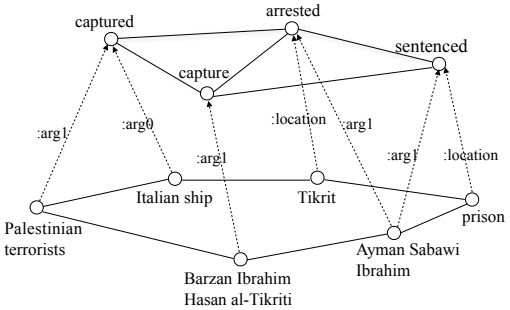

Figure 5: Joint Constraint Clustering for E3,4,5.

Given a trigger set T and their corresponding argument set A, we compute the similarity between two triggers t_1 and t_2 and two arguments a_1 and a_2 by:

$$sim(t_1, t_2) = \lambda \cdot sim_{\cos}(E_g^{t_1}, E_g^{t_2}) + (1-\lambda) \cdot \frac{\Sigma_{r \in R_{t_1} \cap R_{t_2}} sim_{\cos}(E_r^{t_1}, E_r^{t_2})}{|R_{t_1} \cap R_{t_2}|} + f(t_1, t_2)$$

$$sim(a_1, a_2) = sim_{\cos}(E_g^{a_1}, E_g^{a_2}) + f(a_1, a_2)$$

where E_g^t represents the trigger sense vector and E_g^a is the argument vector. R_t is the AMR relation set in the event structure of t, and E_r^t denotes the vector resulting from the last application of the compositional function corresponding to the semantic relation r for trigger t. λ is a regularization parameter that controls the trade-off between these two types of representations. In our experiment $\lambda = 0.6$.

We design a joint constraint clustering approach, which iteratively produces new clustering results based on the above constraints. To find a global optimum, which corresponds to an approximately optimal partition of the trigger set into K clusters $\mathcal{C}^T = \{\mathcal{C}_1^T, \mathcal{C}_2^T, ..., \mathcal{C}_K^T\}$, and a partition of the argument set into M clusters $\mathcal{C}^A = \{\mathcal{C}_1^A, \mathcal{C}_2^A, ..., \mathcal{C}_M^A\}$, we minimize the agreement across clusters and the disagreement within clusters:

$$\arg \min_{K_T, K_A, \lambda} O = (D_{\text{inter}}^T + D_{\text{intra}}^T) + (D_{\text{inter}}^A + D_{\text{intra}}^A)$$

$$D_{\text{inter}}^{\mathcal{P}} = \sum_{i \neq j = 1}^{K} \sum_{u \in \mathcal{C}_i^{\mathcal{P}}, v \in \mathcal{C}_j^{\mathcal{P}}} sim(\mathcal{P}_u, \mathcal{P}_v)$$

$$D_{\text{intra}}^{\mathcal{P}} = \sum_{i=1}^{K} \sum_{u,v \in \mathcal{C}_i^{\mathcal{P}}} (1 - sim(\mathcal{P}_u, \mathcal{P}_v))$$

We incorporate the Spectral Clustering algorithm (Luxburg, 2007) into joint constraint clustering process to get the final optimized clustering results. The detailed algorithm is summarized in Algorithm 1.

2.6 Event Type and Argument Role Naming

For each trigger cluster, we utilize the trigger which is nearest to the centroid of the cluster as the event type name. For a given event trigger, we assign a role name to each of its arguments (identified as in section 2.2). This process depends on which meaning representation was used to select the arguments.

For AMR, we first map the event trigger's OntoNotes sense to PropBank, VerbNet, and FrameNet. We assign each argument a role name as follows. We map AMR core roles (e.g. *":ARG0", "ARG1"*) to FrameNet if possible, otherwise to VerbNet if possible, and finally to PropBank roles if a mapping to VerbNet is not available.[5]. Nearly 5% of AMR core roles can

[5]OntoNotes 5.0 provides a mapping; https://catalog.ldc.upenn.edu/LDC2013T19

Algorithm 1 Joint Constraint Clustering Algorithm

Input: Trigger set T, argument set A, their lexical embedding E_g^T, E_g^A, event structure representation E_R^T, and the minimal (K_T^{min}, K_A^{min}) and maximal (K_T^{max}, K_A^{max}) number of clusters for triggers and arguments;
Output: The optimal clustering results: $\mathcal{C}^T$ and $\mathcal{C}^A$;

- $O_{min} = \infty, \mathcal{C}^T = \varnothing, \mathcal{C}^A = \varnothing$
- For $K_T = K_T^{min}$ to $K_T = K_T^{max}$, $K_A = K_A^{min}$ to $K_A = K_A^{max}$
  - Clustering with Spectral Clustering Algorithm:
  - $\mathcal{C}_{curr}^T = spectral(T, E_g^T, E_R^T, K_T)$
  - $\mathcal{C}_{curr}^A = spectral(A, E_g^A, K_A)$
  - $O_{curr} = O(\mathcal{C}_{curr}^T, \mathcal{C}_{curr}^A)$
  - if $O_{curr} < O_{min}$
    - $O_{min} = O_{curr}, \mathcal{C}^T = \mathcal{C}_{curr}^T, \mathcal{C}^A = \mathcal{C}_{curr}^A$
  - while iterate time ≤ 10
    - $\mathcal{C}_{curr}^T = spectral(T, E_g^T, E_R^T, K_T, \mathcal{C}_{curr}^A)$
    - $\mathcal{C}_{curr}^A = spectral(A, E_g^A, K_A, \mathcal{C}_{curr}^T)$
    - $O_{curr} = O(\mathcal{C}_{curr}^T, \mathcal{C}_{curr}^A)$
    - if $O_{curr} < O_{min}$
      - O_{min}=$O_{curr}, \mathcal{C}^T = \mathcal{C}_{curr}^T, \mathcal{C}^A = \mathcal{C}_{curr}^A$
- return $O_{min}, \mathcal{C}^T, \mathcal{C}^A$;

be mapped to FrameNet roles and 55% can be mapped to VerbNet roles, and the remaining can be mapped to PropBank. Table 3 shows some mapping examples. We map non-core roles from AMR to FrameNet, as shown in Table 4.

When Stanford Typed Dependencies are used for meaning representation we construct a manual mapping AMR relations and use the above procedure. When FrameNet is used for meaning representation we simply keep the FrameNet role name for argument role naming.

Concept	AMR Core Role	FrameNet Role	VerbNet Role	PropBank Description
fire.1	ARG0	Agent	Agent	Shooter
fire.1	ARG1	Projectile	Theme	Gun/projectile
extrude.1	ARG0		Agent	Extruder, agent
extrude.1	ARG1		Theme	Entity extruded
extrude.1	ARG2		Source	Extruded from
blood.1	ARG0			Agent
blood.1	ARG1			Theme, one bled

Table 3: Core Role Mapping Examples.

3 Evaluation

3.1 Data

We used the August 11, 2014 English Wikipedia dump to learn trigger sense and argument embeddings. For evaluation we choose a subset of ERE (Entity Relation Event) corpus (50 documents) which has perfect AMR annotations so we can compare the impact of perfect AMR and system generated AMR. To compare with state-of-the-art event extraction on Automatic Content Extraction (ACE2005) data, we follow the same evaluation setting in previous work (Ji and Grishman, 2008; Liao and Grishman, 2010; Hong et al., 2011) and use 40 newswire documents as our test set.

AMR None-Core Role	FrameNet Role
topic	Topic
instrument	Instrument
manner	Manner
poss	Possessor
prep-for, prep-to, prep-on-behalf	Purpose
time, decade, year, weekday, duration	Time
mod, cause, prep-as	Explanation
prep-by, medium, path	Means
location, destination, prep-in	Place

Table 4: None-Core Role Mapping.

3.2 Schema Discovery

Figure 6 shows some examples as part of the event schema discovered from the ERE data set. Each cluster denotes an event type, with a set of event mentions and sentences. Each event mention is also associated with some arguments and their roles. The event and argument role annotations for sample sentences may serve as an example-based corpus-customized "annotation guideline" for event extraction.

Table 5 compares the coverage of event schema discovered by our approach, using AMR as meaning representation, with the predefined ACE and ERE event schemas. Besides the types defined in ACE and ERE, this approach discovers many new event types such as **Build** and **Threaten** as displayed in Figure 6. Our approach can also discover new argument roles for a given event type. For example, for Attack events, besides five types of existing arguments (Attacker, Target, Instrument, Time, and Place) defined in ACE, we also discover a new type of argument **Purpose**. For example, in "*The Dutch government, facing strong public anti-war pressure, said it would not commit fighting forces to the war against Iraq but added it supported the military* ***campaign*** *to* ***disarm Saddam***.", "*disarm Saddam*" is identified as the **Purpose** for the **Attack** event triggered by "*campaign*". Note that while FrameNet specifies Purpose as an argument role for the Attack, such information specific to Attack is not part of AMR.

Event Type: Transport

S1: The court official stated that on 18 March 2008 Luong stated to judges that she was hired by an unidentified *man* to **ship** the *heroin* to *Australia* in exchange for 15000 U.S. dollars.
***Event**: ship,*
***Arguments**: man(Agent), Austrialia(Destination),heroin(Theme)*

S2: State media didn't identify the 2 convicts hanged in Zahedan but stated that *they* had been found guilty of **transporting** 5.25 kilograms of *heroin*.
***Event**: transporting, **Arguments**: they(Agent), heroin(Theme)*

Event Type: Die

S1: Police in the strict communist country discovered his methamphetamine manufacturing plant disguised as a soap factory and sentenced *him* to **death** in *1997*.
***Event**: death, **Arguments**: him(Theme), 1997(Time)*

S2: A newspaper report on January 1, 2008 that *Iran* **hanged** two convicted *drug traffickers* in the *south-eastern city of Zahedan*.
***Event**: hanged, **Arguments**: Iran(Agent), drug traffickers(Theme), southeastern city of Zahedan(Place)*

Event Type: Build

S1: The **construction** of the *facility* started in *790000*, but stopped after the 910000 Soviet collapse when Tajikistan slid into a 5 year civil war that undermined its economy.
***Event**:construction,*
***Arguments**: facility(Product), 790000(Time)*

S2: The closed *Soviet-era military facility* was **fou-nded** in *570000* and collects and analyzes all information gathered from Russia's military spy satellites.
***Event**: founded,*
***Arguments**: Soviet-era facility(Product), 570000(Time)*

Event Type: Threaten

S1: *Colombian Government* was **alarmed** because uranium is the primary basis for generating weapons of mass destruction.
***Event**:alarmed, **Arguments**:Columbian Government(Experiencer)*

S2: Cluster bomblets have been criticized by human rights groups because they kill indiscriminately and because unexploded *ordinance* poses a **threat** to *civilians* similar to that of land mines.
***Event**:threat,*
***Arguments**:ordinance(Cause), civilian(Experiencer)*

Figure 6: Example Output of the Event Schema.

Data	ACE			ERE				
	Human	*SystemAMR*	*Overlap*	*Human*	*PerfectAMR*	*Overlap*	*SystemAMR*	*Overlap*
# of Events	440	2,395	331	580	3,765	517	2,498	477
# of Event Types	33	134	N/A	26	137	N/A	120	N/A
# of Arguments	883	4,361	587	1,231	6,195	919	4,288	801

Table 5: Schema Coverage Comparison on ACE and ERE.

3.3 Event Extraction for All Types

To evaluate the performance of the whole event schema, we randomly sample 100 sentences from ERE data set and ask two linguistic experts to fully annotate the events and arguments. As a starting point, annotators were given output from our Schema Discovery using gold standard AMR. For each sentence, they saw event triggers and corresponding arguments. Their job was to correct this output by marking incorrectly identified events and arguments, and adding missing events and arguments. The inter-annotator agreement is 83% for triggers and 79% for arguments.

To evaluate trigger and argument identification, we automatically compare this gold standard with system output (see Table 6). To evaluate trigger and argument typing, annotators manually checked system output and assessed whether the type name was reasonable (see Table 6). Note that automatic comparison between system and gold standard output is not appropriate for typing; for a given cluster, there is no definitive "best" name.

We found that most event triggers not recovered by our system are multi-word expressions such as "*took office*" or adverbs such as "*previously*" and "*formerly*". For argument identification, our approach fails to identify some arguments that require world knowledge to extract. For example, in "*Anti-corruption judge **Saul Pena** stated Montesinos has admitted to the abuse of authority **charge***", "*Saul Pena*" is not identified as a **Adjudicator** argument of event "*charge*" because it has no direct semantic relations with the event trigger.

3.4 Impact of Semantic Information and Meaning Representations

Table 7 assesses the impact of various types of semantic information, and also compares the effectiveness of each type of meaning representation for the typing task only. We note that F-measure drops 14.4 points if only WSD based embeddings are not used. In addition, AMR relations specifying both core and non-core roles are informative for learning distinct compositional operators. To compare typing results across meaning representations, we use triggers identified by both the AMR and FrameNet parsers. Using Stanford Typed Dependencies, relations are likely too coarse-grained or lack sufficient semantic information. Thus, our approach cannot leverage the inter-dependency between event trigger type and argument role to achieve pure trigger clusters. Compared with dependency relations, the fine-grained AMR semantic relations such as *:location*, *:manner*, *:topic*, *:instrument* appear to be more informative to infer the argument roles. For example, in sentence "*Approximately 25 kilometers southwest of Sringar 2 militants were killed in a second gun battle.*", "*gun*" is identified as an **Instrument** for "*battle*" event based on the AMR relation :instrument. In contrast, dependency parsing identifies "*gun*" as a

Method	Trigger Identification (%)			Trigger Typing (%)			Arg Identification (%)			Arg Typing (%)		
	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1
Perfect AMR	87.0	98.7	92.5	70.0	79.5	74.5	94.0	83.7	88.6	72.4	64.4	68.2
System AMR	93.0	67.2	78.0	69.8	50.5	58.6	95.7	59.6	73.4	68.9	42.9	52.9

Table 6: Overall Performance of Liberal Event Extraction on ERE data for All Event Types.

Method	Trigger F_1 (%)			Arg F_1 (%)		
	P	R	F_1	P	R	F_1
Perfect AMR	70	79.5	74.5	72.4	64.4	68.2
w/o Structure Representation	52.8	59.4	55.9	52.1	48.0	50.0
w/o WSD based embeddings	62.8	57.4	60.1	61.9	50.3	55.5
w/o None-Core Roles	61.5	72.2	66.5	61.3	58.0	59.6
w/o Core Roles	57.3	49.7	53.2	63.6	49.5	55.7
System AMR	69.8	50.5	58.6	68.9	42.9	52.9
Replace AMR with Dependency Parsing	45.9	61.9	52.7	63.9	18.2	28.4
Replace AMR with FrameNet Parsing	43.1	57.1	49.2	78.1	7.1	13.0

Table 7: Impact of semantic information and representations on typing for ERE data.

compound modifier of "*battle*". Note that we used a static mapping to map dependency relations to AMR relations (see section 2.6), whereas ideally this mapping would be context-dependent. Creating a context-dependent mapping would constitute significant steps toward building an AMR parser.

Using FrameNet results in low recall for argument typing. SEMAFOR's output often does not identify all the arguments identified by our annotators. Many triggers are associated with zero or one argument, thus there is not enough data to learn the event structure representation. In addition, most of the arguments from identified by SEMAFOR are long phrases. Because no internal structure is assigned, we simply average all single token's vectors to represent the phrase. However, the high precision may be due to the fact that FrameNet relations are designed to specify semantic roles.

3.5 Event Extraction for ACE/ERE Types

We manually select the event triggers in the ACE and ERE evaluation sets discovered by our AMR-based approaches that are ACE/ERE events based on their annotation guidelines. If a trigger doesn't already have a gold standard ACE/ERE annotation we provide one. For each such event we use core roles and Instrument/Possessor/Time/Place relations to detect arguments. Each trigger and argument role type is assessed manually if an ACE/ERE annotation does not exist. We evaluate our approach for trigger and argument typing by comparing system output to manual annotation, considering synonymous labels to be equivalent (e.g., our approach's *kill* type ACE's *die*). We compare our approach with the following state-of-the-art supervised methods which are trained from 529 ACE documents or 336 ERE documents:

- DMCNN: A dynamic multi-pooling convolutional neural network based on distributed word representations (Chen et al., 2015).
- Joint: A structured perceptron model based on symbolic semantic features (Li et al., 2013).
- LSTM: A long short-term memory neural network (Hochreiter and Schmidhuber, 1997) based on distributed semantic features.

Table 8 shows the results. On ACE events, both DMCNN and Joint methods outperform our approach for trigger and argument extraction. However, when moving to ERE event schema, although re-trained based on ERE labeled data, their performance still degrades significantly. These previous methods heavily rely on the quality and quantity of the training data. When the training data is not adequate (the ERE training documents contain 1,068 events and 2,448 arguments, while ACE training documents contain more than 4,700 events and 9,700 arguments), the performance is low. In contrast, our approach is unsupervised and can automatically identify events, arguments and assign types/roles, and is not tied to one event schema.

3.6 Event Extraction for Biomedical Domain

To demonstrate the portability of our approach to a new domain, we conduct our experiment on 14 biomedical articles (755 sentences) with perfect AMR annotations (Garg et al., 2016). We utilize a word2vec model[6] trained from all paper abstracts from PubMed[7] and full-text documents from the PubMed Central Open Access subset. To evaluate the performance, we randomly sample 100 sentences and ask a biomedical scientist to assess the correctness of each event and argument role. Our approach achieves 83.1% precision on trigger labeling (619 events in total) and 78.4% precision on argument labeling (1,124 arguments in total).

[6] http://bio.nlplab.org/

[7] http://www.ncbi.nlm.nih.gov/pubmed

Method	ERE: Trigger F_1 (%)			ERE: Arg F_1(%)			ACE: Trigger F_1 (%)			ACE: Arg F_1 (%)		
	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1
LSTM	41.5	46.8	44.1	9.9	11.6	10.7	66.0	60	62.8	29.3	32.6	30.8
Joint	42.3	41.7	42.0	61.8	23.2	33.7	73.7	62.3	67.5	64.7	44.4	52.7
DMCNN	-	-	-	-	-	-	75.6	63.6	**69.1**	68.8	46.9	**53.5**
Liberal$_{PerfectAMR}$	79.8	50.5	**61.8**	48.9	32.9	**39.3**	-	-	-	-	-	-
Liberal$_{SystemAMR}$	88.5	42.6	57.5	47.6	30.0	36.8	80.7	50.1	61.8	51.9	39.4	44.8

Table 8: Performance on ERE and ACE events.

It demonstrates that our approach can be rapidly adapted to a new domain and discover domain-rich event schema. An example schema for an event type "*Dissociate*" is shown in Figure 7.

Event Type: Dissociate

S1: Ras acts as a molecular switch that is activated upon GTP loading and deactivated upon **hydrolysis** of GTP to GDP.

***Event**: hydrolysis **Arguments**:GTP (Patient), (GDP) (Result)*

S2: Activation requires **dissociation** of protein-bound GDP , an intrinsically slow process that is accelerated by guanine nucleotide exchange factors.

***Event**: dissociation **Arguments**: GDP (Patient)*

S3: His - ubiquitinated proteins were purified by Co2+ metal affinity chromatography in 8M urea **denaturing** conditions.

***Event**: denaturing **Arguments**: proteins(Patient)*

Figure 7: Example Output of the Discovered Biomedical Event Schema.

4 Related Work

Most of previous event extraction work focused on learning supervised models based on symbolic features (Ji and Grishman, 2008; Miwa et al., 2009; Liao and Grishman, 2010; Liu et al., 2010; Hong et al., 2011; McClosky et al., 2011; Sebastian and Andrew, 2011; Chen and Ng, 2012; Li et al., 2013) or distributional features through deep learning (Chen et al., 2015; Nguyen and Grishman, 2015). They usually rely on a predefined event schema and a large amount of training data. Compared with other paradigms such as Open Information Extraction (Etzioni et al., 2005; Banko et al., 2007; Banko et al., 2008; Etzioni et al., 2011; Ritter et al., 2012), Preemptive IE (Shinyama and Sekine, 2006), On-demand IE (Sekine, 2006) and semantic frame based event discovery (Kim et al., 2013), our approach can explicitly name each event type and argument role. Some recent work focused on universal schema discovery (Chambers and Jurafsky, 2011; Pantel et al., 2012; Yao et al., 2012; Yao et al., 2013; Chambers, 2013; Nguyen et al., 2015). However, the schemas discovered from these methods are rather static and they are not customized for any specific input corpus.

Our work is also related to efforts at composing word embeddings using syntactic structures (Hermann and Blunsom, 2013; Socher et al., 2013a; Socher et al., 2013b; Bowman et al., 2014; Zhao et al., 2015). Our trigger sense representation is similar to Word Sense Induction (Navigli, 2009; Bordag, 2006; Pinto et al., 2007; Brody and Lapata, 2009; Manandhar et al., 2010; Navigli and Lapata, 2010; Van de Cruys and Apidianaki, 2011; Wang et al., 2015b). Besides word sense, we exploit related concepts to enrich trigger representation.

5 Conclusions and Future Work

We proposed a novel Liberal event extraction framework which combines the merits of symbolic semantics and distributed semantics. Experiments on news and biomedical domain demonstrate that this framework can discover explicitly defined rich event schemas which cover not only most types in existing manually defined schemas, but also new event types and argument roles. The granularity of event types is also customized for specific input corpus. And it can produce high-quality event annotations simultaneously without using annotated training data. In the future, we will extend this framework to other Information Extraction tasks.

Acknowledgements

We would like to thank Kevin Knight and Jonathan May (ISI) for sharing biomedical AMR annotations. This work was supported by the U.S. ARL NS-CTA No. W911NF-09-2-0053 and DARPA DEFT No. FA8750-13-2-0041, and in part by NSF IIS-1523198, IIS-1017362, IIS-1320617 and IIS-1354329, and NIH BD2K grant 1U54GM114838. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

References

C. F. Baker and H. Sato. 2003. The framenet data and software. In *Proc. ACL2003*.

L. Banarescu, C. Bonial, S. Cai, M. Georgescu, K. Griffitt, U. Hermjakob, K. Knight, P. Koehn, M. Palmer, and N. Schneider. 2013. Abstract meaning representation for sembanking. In *Proc. ACL2013 Workshop on Linguistic Annotation and Interoperability with Discourse*.

M. Banko, M. Cafarella, S. Soderland, M. Broadhead, and O. Etzioni. 2007. Open information extraction for the web. In *Proc. IJCAI2007*.

M. Banko, O. Etzioni, and T. Center. 2008. The tradeoffs between open and traditional relation extraction. In *Proc. ACL-HLT2008*.

S. Bordag. 2006. Word sense induction: Triplet-based clustering and automatic evaluation. In *Proc. EACL2006*.

S. Bowman, C. Potts, and C. Manning. 2014. Recursive neural networks for learning logical semantics. *CoRR*, abs/1406.1827.

S. Brody and M. Lapata. 2009. Bayesian word sense induction. In *Proc. EACL2009*.

N. Chambers and D. Jurafsky. 2011. Template-based information extraction without the templates. In *Proc. ACL-HLT2011*.

N. Chambers. 2013. Event schema induction with a probabilistic entity-driven model. In *EMNLP*, volume 13, pages 1797–1807.

C. Chen and V. Ng. 2012. Joint modeling for chinese event extraction with rich linguistic features. In *In COLING*. Citeseer.

Y. Chen, L. Xu, K. Liu, D. Zeng, and J. Zhao. 2015. Event extraction via dynamic multi-pooling convolutional neural networks. In *Proc. ACL2015*.

Dipanjan Das, Desai Chen, André FT Martins, Nathan Schneider, and Noah A Smith. 2014. Frame-semantic parsing. *Computational Linguistics*, 40(1):9–56.

O. Etzioni, M. Cafarella, D. Downey, A. Popescu, T. Shaked, S. Soderland, D. Weld, and A. Yates. 2005. Unsupervised named-entity extraction from the web: An experimental study. *Artificial Intelligence*, 165:91–134.

O. Etzioni, A. Fader, J. Christensen, S. Soderland, and M. Mausam. 2011. Open information extraction: The second generation. In *Proc. IJCAI2011*, volume 11, pages 3–10.

S. Garg, A. Galstyan, U. Hermjakob, and D. Marcu. 2016. Extracting biomolecular interactions using semantic parsing of biomedical text. In *Proc. AAAI*.

R. Grishman and B. Sundheim. 1996. Message understanding conference-6: A brief history. In *Proc. COLING1996*.

Z. Harris. 1954. Distributional structure. *Word*, 10(23):146–162.

K. Hermann and P. Blunsom. 2013. The role of syntax in vector space models of compositional semantics. In *Proc. ACL2013*.

S. Hochreiter and J. Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Y. Hong, J. Zhang, B. Ma, J. Yao, G. Zhou, and Q. Zhu. 2011. Using cross-entity inference to improve event extraction. In *Proc. ACL*, pages 1127–1136. Association for Computational Linguistics.

E. Hovy, M. Marcus, M. Palmer, L. Ramshaw, and R. Weischedel. 2006. Ontonotes: the 90% solution. In *Proc. NAACL2006*.

H. Ji and R. Grishman. 2008. Refining event extraction through cross-document inference. In *ACL*.

H. Ji and R. Grishman. 2011. Knowledge base population: Successful approaches and challenges. In *Proc. ACL2011*.

H. Kim, X. Ren, Y. Sun, C. Wang, and J. Han. 2013. Semantic frame-based document representation for comparable corpora. In *ICDM*.

K. Kipper, A. Korhonen, N. Ryant, and M. Palmer. 2008. A large-scale classification of english verbs. *Language Resources and Evaluation Journal*, 42(1):21–40.

Q. Li, H. Ji, and L. Huang. 2013. Joint event extraction via structured prediction with global features. In *Proc. ACL*, pages 73–82.

S. Liao and R. Grishman. 2010. Using document level cross-event inference to improve event extraction. In *Proc. ACL*.

B. Liu, L. Qian, H. Wang, and G. Zhou. 2010. Dependency-driven feature-based learning for extracting protein-protein interactions from biomedical text. In *Proc. COLING*.

U. Luxburg. 2007. A tutorial on spectral clustering. *Statistics and computing*, 17(4):395–416.

S. Manandhar, I. Klapaftis, D. Dligach, and S. Pradhan. 2010. Semeval-2010 task 14: Word sense induction & disambiguation. In *Proc. ACL2010 international workshop on semantic evaluation*.

Dan Klein Christopher D Manning. 2003. Natural language parsing. In *Advances in Neural Information Processing Systems 15: Proceedings of the 2002 Conference*, volume 15, page 3. MIT Press.

D. M. Marie-Catherine, Bill M., and Christopher D. M. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings LREC*, pages 449,454.

D. McClosky, M. Surdeanu, and C. D. Manning. 2011. Event extraction as dependency parsing. In *ACL*, pages 1626–1635.

T. Mikolov, K. Chen, G. Corrado, and J. Dean. 2013. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781.

M. Miwa, R. Stre, Y. Miyao, and J. Tsujii. 2009. A rich feature vector for protein-protein interaction extraction from multiple corpora. In *Proc. EMNLP*.

R. Navigli and M. Lapata. 2010. An experimental study of graph connectivity for unsupervised word sense disambiguation. *Pattern Analysis and Machine Intelligence*, 32(4):678–692.

R. Navigli. 2009. Word sense disambiguation: A survey. *ACM Computing Surveys (CSUR)*, 41(2):10.

T. Nguyen and R. Grishman. 2015. Event detection and domain adaptation with convolutional neural networks. *Volume 2: Short Papers*, page 365.

K. Nguyen, X. Tannier, O. Ferret, and R. Besançon. 2015. Generative event schema induction with entity disambiguation. In *Proc. ACL*.

B. Onyshkevych, M. E. Okurowski, and L. Carlson. 1993. Tasks, domains, and languages for information extraction. In *TIPSTER*.

M. Palmer, D. Gildea, and P. Kingsbury. 2005. The proposition bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–106.

P. Pantel, T. Lin, and M. Gamon. 2012. Mining entity types from query logs via user intent modeling. In *Proc. ACL2012*.

D. Pinto, P. Rosso, and H. Jimenez-Salazar. 2007. Upv-si: Word sense induction using self term expansion. In *Proc. ACL2007 International Workshop on Semantic Evaluations*.

S. Pradhan, L. Ramshaw, M. Marcus, M. Palmer, R. Weischedel, and N. Xue. 2011. Conll-2011 shared task: Modeling unrestricted coreference in ontonotes. In *Proc. CONLL2011*.

A. Ritter, O. Etzioni, and S. Clark. 2012. Open domain event extraction from twitter. In *Proc. SIGKDD2012*, pages 1104–1112. ACM.

R. Sebastian and M. Andrew. 2011. Fast and robust joint models for biomedical event extraction. In *EMNLP*.

S. Sekine. 2006. On-demand information extraction. In *Proc. COLING-ACL2006*.

Y. Shinyama and S. Sekine. 2006. Preemptive information extraction using unrestricted relation discovery. In *Proc. HLT-NAACL2006*.

R. Socher, J. Pennington, E. H. Huang, A. Y. Ng, and C. D. Manning. 2011. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proc. EMNLP*, pages 151–161.

R. Socher, A. Karpathy, Q. V. Le, C. Manning, and A. Y. Ng. 2013a. Grounded compositional semantics for finding and describing images with sentences. *TACL2013*.

R. Socher, A. Perelygin, J. Wu, J. Chuang, C. Manning, A. Ng, and C. Potts. 2013b. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proc. EMNLP2013*.

E. Tjong, K. Sang, and F. Meulder. 2003. Introduction to the conll-2003 shared task: language-independent named entity recognition. In *Proc. CONLL2003*.

T. Van de Cruys and M. Apidianaki. 2011. Latent semantic word sense induction and disambiguation. In *Proc. ACL-HLT2011*.

C. Wang, N. Xue, and S. Pradhan. 2015a. Boosting transition-based amr parsing with refined actions and auxiliary analyzers. In *Proc. ACL2015*.

J. Wang, M. Bansal, K. Gimpel, B. Ziebart, and T. Clement. 2015b. A sense-topic model for word sense induction with unsupervised data enrichment. *TACL*, 3:59–71.

L. Yao, S. Riedel, and A. McCallum. 2012. Probabilistic databases of universal schema. In *Proc. NIPS2012 Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction*.

L. Yao, S. Riedel, and A. McCallum. 2013. Universal schema for entity type prediction. In *Proc. NIPS2013 Workshop on Automated Knowledge Base Construction*.

Y. Zhao, Z. Liu, and M. Sun. 2015. Phrase type sensitive tensor indexing model for semantic composition. In *Proc. AAAI2015*.

Z. Zhong and H. T. Ng. 2010. It makes sense: A wide-coverage word sense disambiguation system for free text. In *Proceedings of the ACL 2010 System Demonstrations*, pages 78–83.