Seed-Based Event Trigger Labeling: How far can event descriptions get us?

Ofer Bronstein[1], **Ido Dagan**[1], **Qi Li**[2], **Heng Ji**[2], **Anette Frank**[3,4]
[1] Computer Science Department, Bar-Ilan University
[2] Department of Computer Science, Rensselaer Polytechnic Institute
[3] Department of Computational Linguistics, Heidelberg University
[4]Research Training Group AIPHES, Dept. of Computational Linguistics, Heidelberg University
oferbr@gmail.com dagan@cs.biu.ac.il
{liq7,jih}@rpi.edu frank@cl.uni-heidelberg.de

Abstract

The task of event trigger labeling is typically addressed in the standard supervised setting: triggers for *each* target event type are annotated as training data, based on annotation guidelines. We propose an alternative approach, which takes the example trigger terms mentioned in the guidelines as seeds, and then applies an event-independent similarity-based classifier for trigger labeling. This way we can skip manual annotation for new event types, while requiring only minimal annotated training data for few example events at system setup. Our method is evaluated on the ACE-2005 dataset, achieving 5.7% F_1 improvement over a state-of-the-art supervised system which uses the full training data.

1 Introduction

Event trigger labeling is the task of identifying the main word tokens that express mentions of pre-specified event types in running text. For example, in "20 people were *wounded* in Tuesday's airport *blast*", "*wounded*" is a trigger of an *Injure* event and "*blast*" is a trigger of an *Attack*. The task both detects trigger tokens and classifies them to appropriate event types. While this task is often a component within the broader *event extraction* task, labeling both triggers and arguments, this paper focuses on trigger labeling.

Most state-of-the-art event trigger labeling approaches (Ji and Grishman, 2008; Liao and Grishman, 2010b; Hong et al., 2011; Li et al., 2013) follow the standard supervised learning paradigm. For each event type, experts first write annotation guidelines. Then, annotators follow them to label event triggers in a large dataset. Finally, a classifier is trained over the annotated triggers to label the target events.

The supervised paradigm requires major human efforts both in producing high-quality guidelines and in dataset annotation for each new event type. Given the rich information embedded in the guidelines, we raise in this paper the following research question: how well can we perform by leveraging *only* the lexical knowledge already available in quality guidelines for *new* event types, without requiring annotated training data for them?

To address this question, we propose a seed-based approach for the trigger labeling task (Section 2). Given the description for a new event type, which contains some examples of triggers, we first collect these triggers into a list of *seeds*. Then, at the labeling phase, we consider each text token as a candidate for a trigger and assess its similarity to the seed list. In the above example, given seeds such as "*explosion*" and "*fire*" for the *Attack* event type, we identify that the candidate token "*blast*" is a hyponym of "*explosion*" and synonym of "*fire*" and infer that "*blast*" is a likely *Attack* trigger.

In our method, such similarity indicators are encoded as a small set of event-independent classification features, based on lexical matches and external resources like WordNet. Using event-independent features allows us to train the system only once, at system setup phase, requiring annotated triggers in a training set for just a few pre-selected event types. Then, whenever a new event type is introduced for labeling, we only need to collect a seed list for it from its description, and provide it as input to the system.

We developed a seed-based system (Section 3), based on a state-of-the-art fully-supervised event extraction system (Li et al., 2013). When evaluated on the ACE-2005 dataset,[1] our system outperforms the fully-supervised one (Section 4), even though we don't utilize any annotated triggers of the test events during the labeling phase, and only

[1]http://projects.ldc.upenn.edu/ace

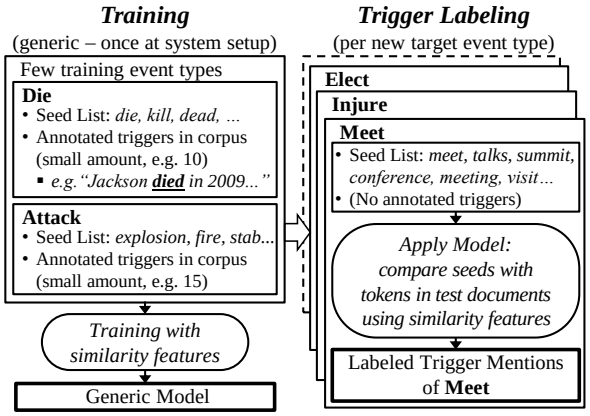

Figure 1: Flow of the seed-based approach

use the seed triggers appearing in the ACE annotation guidelines. This result contributes to the broader line of research on avoiding or reducing annotation cost in information extraction (Section 5). In particular, it suggests the potential utility of the seed-based approach in scenarios where manual annotation per each new event is too costly.

2 Seed-Based Problem Setup

This section describes our setup, as graphically illustrated in Figure 1.

Similarly to the supervised setting, our approach assumes that whenever a new event type is defined as target, an informative *event description* should be written for it. As a prominent example, we consider Section 5 of the ACE-2005 event annotation guidelines,[2] which provides a description for each event type. The description includes a short verbal specification plus several illustrating example sentences with marked triggers, spanning on average less than a page per event type.

As event descriptions specify the intended event scope, they inherently include representative examples for event triggers. For instance, the ACE-2005 guidelines include: *"MEET Events include talks, summits, conferences, meetings, visits,..."*, followed by an example: *"Bush and Putin met this week..."*. We thus collect triggers mentioned in each event description into a *seed list* for the event type, which is provided as input to our trigger labeling method. Triggers from the above quoted sentences are hence included in the *Meet* seed list, shown in Figure 1.

As mentioned in the Introduction, our method (Section 3) is based on event-independent features that identify similarities between a candidate trigger and a given seed list. To train such generic features, our training requires several arbitrary *training event types*, with a small amount of annotated triggers, from which it learns weights for the features. In our evaluation (Section 4) we use 5 training event types, with a total of 30 annotated trigger mentions (compared to roughly 5000 used by the baseline fully-supervised system). In this setting, the training phase is required only once during system setup, while no further training is required for each new target event type.

In summary, our setup requires: (1) a seed list per target event type; (2) a small number of annotated triggers for few training event types, along with their seed lists (at system setup).

3 Method

This section describes the method we designed to implement the *seed-based* approach. To assess our approach, we compare it (Section 4) with the common *fully-supervised* approach, which requires annotated triggers for each target event type. Therefore, we implemented our system by adapting the state-of-the-art fully-supervised event extraction system of Li et al. (2013), modifying mechanisms relevant for features and for trigger labels, as described below. Hence the systems are comparable with respect to using the same preprocessing and machine learning infrastructure.

3.1 The Fully-Supervised System

The event extraction system of Li et al. (2013) labels triggers and their arguments for a set of target event types $\mathcal{L}$, for which annotated training documents are provided. The system utilizes a structured perceptron with beam search (Collins and Roark, 2004; Huang et al., 2012). To label triggers, the system scans each sentence x, and creates candidate assignments y, that for each token x_i assign each possible label $y_i \in \mathcal{L} \cup \{\perp\}$ ($\perp$ meaning x_i is not a trigger at all). The score of an assignment (x_i, y_i) is calculated as $\mathbf{w} \cdot \mathbf{f}$, where $\mathbf{f}$ is the binary feature vector calculated for (x_i, y_i), and $\mathbf{w}$ is the learned feature weight vector.

The classifier's features capture various properties of x_i and its context, such as its unigram and its containing bigrams. These features are highly lexicalized, resulting in a very large feature space. Additionally, each feature is replicated and paired with each label y_i, allowing the system to learn

[2] https://www.ldc.upenn.edu/sites/www.ldc.upenn.edu/files/english-events-guidelines-v5.4.3.pdf

Feature	Description
Same Lemma	Do the candidate token and a seed share the same lemma?
Synonym	Is a seed a WN synonym of the candidate token?
Hypernym	Is a seed a WN hypernym or instance-hypernym of the candidate token?
Similarity Relations	Does one of these WN relations hold between a seed and a candidate token? Synonym, Hypernym, Instance Hypernym, Part Holonym, Member Holonym, Substance Meronym, Entailment

Table 1: Similarity features using WordNet (WN). For the last two features we allow up to 2 levels of transitivity (e.g. hypernym of hypernym), and consider also derivations of candidate tokens.

different weights for different labels, e.g., feature *(Unigram:"visited", Meet)* will have a different weight than *(Unigram:"visited", Attack)*.

3.2 The Seed-Based System

To implement the seed-based approach for trigger labeling, we adapt only the trigger classification part in the Li et al. (2013) fully-supervised system, ignoring arguments. Given a set of new target event types $\mathcal{T}$ we classify every test sentence once for each event type $t \in \mathcal{T}$. Hence, when classifying a sentence for t, the labeling of each token x_i is binary, where $y_i \in \{\top, \bot\}$ marks whether x_i is a trigger of type t ($\top$) or not ($\bot$). For instance x_i="visited" labeled as $\top$ when classifying for *t=Meet*, means x_i is labeled as a *Meet* trigger. To score the binary label assignment (x_i, y_i), we use a small set of features that assess the similarity between x_i and t's given seed list.

We implement our approach with a basic set of binary features (Table 1), which are turned on if similarity is found for at least one seed in the list. We use a single knowledge resource (WordNet (Fellbaum, 1998)) for expansion.[3] As in the fully-supervised system, each feature is replicated for each label in $\{\top, \bot\}$, learning separately how well a feature can predict a trigger ($\top$) and a non-trigger ($\bot$). As labels are event-independent, features are event-independent as well, and their weights can be learned generically (Figure 1).

Since we label each token independently for each event type t, multiple labels may be assigned to the same token. If a single-label setting is required, standard techniques can be applied, such as choosing a single random label, or the highest scoring one.

[3]This could be potentially extended, e.g. with paraphrase databases, like (Ganitkevitch et al., 2013).

4 Evaluation

4.1 Setting

We evaluate our seed-based approach (Section 2) in comparison to the fully-supervised approach implemented by Li et al. (2013) (Section 3). To maintain comparability, we use the ACE-2005 documents with the same split as in (Ji and Grishman, 2008; Liao and Grishman, 2010b; Li et al., 2013) to 40 test documents and 559 training documents. However, some evaluation settings differ: Li et al. (2013) train a multi-class model for all 33 ACE-2005 event types, and classify all tokens in the test documents into these event types. Our approach, on the other hand, trains an event-independent binary classifier, while testing on new event types that are different from those utilized for training. We next describe how this setup is addressed in our evaluation.

Per-Event Classification To label the test documents to all 33 event types, we classify each token in the test documents once for each *test event type*.[4]

Training Event Types When we label for a test event type t, we use a model that was trained on different pre-selected training event types. Since we need to label for all event types, we cannot use the same model for testing them all, since then the event types used to train this model could not be tested. Thus, for each t we use a model trained on 5 randomly chosen training event types, different than t.[5] Additionally, to avoid a bias caused by a particular random choice, we build 10 different models, each time choosing a different set of 5 training event types. Then, we label the test documents for t 10 times, once by each model. When measuring performance we compute the average of these 10 runs for each t, as well as the variance within these runs.

Annotated Triggers Training event types require annotated triggers from the training documents. To maintain consistency between different sets of training event types, we use a fixed total of 30 annotated trigger tokens for each set of

[4]To maintain comparability with the single-label classification results of Li et al. (2013), we randomly choose a single label for our classification in the few (7) cases where it yielded two labels for the same token.

[5]Li et al. (2013) internally split the training documents to "train" and "dev". Accordingly, our training event types are split to 3 "train" events and 2 "dev" events (with annotations taken from the "train" and "dev" documents respectively).

	Micro-Avg. (%)			*Var*
	Prec	*Rec*	F_1	*Avg*
Seed-Based	**80.6**	**67.1**	**73.2**	0.04
Li et al. (2013)	73.7	62.3	67.5	-
Ji and Grishman (2008)	67.6	53.5	59.7	-

Table 2: Seed-based performance compared to fully-supervised systems, plus average F_1 variance (%) over the 10 test runs per test event type.

training event types. The amounts of 5 training event types and 30 annotated triggers were chosen to demonstrate that such small amounts, requiring little manual effort at system setup, yield high performance (larger training didn't improve results, possibly due to the small number of features).

Seed Lists To build the seed lists for all event types, we manually extracted all triggers mentioned in Section 5 of the ACE-2005 guidelines, as described in Section 2.[6] This resulted in lists of 4.2 seeds per event type on average, which is fairly small. For comparison, each event type has an average of 46 distinct trigger terms in the training corpus used by the fully-supervised method.

4.2 Results

Table 2 shows our system's precision, recall and F_1,[7] and the average variance of F_1 within the 10 runs of each test event type. The very low variance indicates that the system's performance does not depend much on the choice of training event types.

We compare our system's performance to the published trigger classification results of the baseline system of (Li et al., 2013) (its globally optimized run, when labeling both triggers and arguments). We also compare to the sentence-level system in (Ji and Grishman, 2008) which uses the same dataset split. Our system outperforms the fully-supervised baseline by 5.7% F_1, which is statistically significant (two-tailed Wilcoxon test, $p < 0.05$). This shows that there is no performance hit for the seed-based method on this dataset, even though it does not require any annotated data for new tested events, thus saving costly annotation efforts.

[6]Our seed lists are publicly available for download at: `https://goo.gl/sErDW9`

[7]We report micro-average as typical for this task. Macro-average results are a few points lower for our system and for the system of Li et al. (2013), maintaining similar relative difference.

5 Related Work

Our work contributes to the broader research direction of reducing annotation for information extraction. One such IE paradigm, including Preemptive IE (Shinyama and Sekine, 2006), On-demand IE (Sekine, 2006; Sekine and Oda, 2007) and Open IE (Etzioni et al., 2005; Banko et al., 2007; Banko et al., 2008), focuses on unsupervised relation and event discovery. We, on the other hand, follow the same goal as fully-supervised systems in targeting pre-specified event types, but aim at minimal annotation cost.

Bootstrapping methods (such as (Yangarber et al., 2000; Agichtein and Gravano, 2000; Riloff, 1996; Greenwood and Stevenson, 2006; Liao and Grishman, 2010a; Stevenson and Greenwood, 2005; Huang and Riloff, 2012)) have been widely applied in IE. Most work started from a small set of seed patterns, and repeatedly expanded them from unlabeled corpora. Relying on unlabeled data, bootstrapping methods are scalable but tend to produce worse results (Liao and Grishman, 2010a) than supervised models due to semantic drift (Curran et al., 2007). Our method can be seen complementary to bootstrapping frameworks, since we exploit manually crafted linguistic resources which are more accurate but may not cover all domains and languages.

Our approach is perhaps closest to (Roth et al., 2009). They addressed a different IE task – relation extraction, by recognizing entailment between candidate relation mentions and seed patterns. While they exploited a supervised recognizing textual entailment (RTE) system, we show that using only simple WordNet-based similarity features and minimal training yields relatively high performance in event trigger labeling.

6 Conclusions and Future Work

In this paper we show that by utilizing the information embedded in annotation guidelines and lexical resources, we can skip manual annotation for new event types. As we match performance of a state-of-the-art fully-supervised system over the ACE-2005 benchmark (and even surpass it), we offer our approach as an appealing way of reducing annotation effort while preserving result quality. Future research may explore additional features and knowledge resources, investigate alternative approaches for creating effective seed lists, and extend our approach to argument labeling.

Acknowledgments

This work was partially supported by the European Commission (project EXCITEMENT, FP7 ICT-287923), and the U.S. DARPA DEFT Program No. FA8750-13-2-0041, ARL NS-CTA No. W911NF-09-2-0053, NSF CAREER IIS-1523198.

References

Eugene Agichtein and Luis Gravano. 2000. *Snowball*: extracting relations from large plain-text collections. In *Proc. Fifth ACM International Conference on Digital Libraries*, pages 85–94.

M. Banko, M. Cafarella, S. Soderland, M. Broadhead, and O. Etzioni. 2007. Open information extraction for the web. In *Proc. IJCAI*, pages 2670–2676.

M. Banko, O. Etzioni, and T. Center. 2008. The tradeoffs between open and traditional relation extraction. In *Proc. ACL*, pages 28–36.

Michael Collins and Brian Roark. 2004. Incremental parsing with the perceptron algorithm. In *Proc. ACL*, pages 111–118.

James R Curran, Tara Murphy, and Bernhard Scholz. 2007. Minimising semantic drift with mutual exclusion bootstrapping. In *Proc. PACLING*, pages 172–180.

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.

Christiane Fellbaum, editor. 1998. *WordNet: An Electronic Lexical Database*. The MIT Press.

Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2013. PPDB: The paraphrase database. In *Proc. NAACL-HLT*, pages 758–764.

Mark A. Greenwood and Mark Stevenson. 2006. Improving semi-supervised acquisition of relation extraction patterns. In *Proc. Workshop on Information Extraction Beyond The Document*, pages 29–35.

Yu Hong, Jianfeng Zhang, Bin Ma, Jian-Min Yao, Guodong Zhou, and Qiaoming Zhu. 2011. Using cross-entity inference to improve event extraction. In *Proc. ACL*, pages 1127–1136.

Ruihong Huang and Ellen Riloff. 2012. Bootstrapped training of event extraction classifiers. In *Proc. ACL*, pages 286–295.

Liang Huang, Suphan Fayong, and Yang Guo. 2012. Structured perceptron with inexact search. In *Proc. NAACL*, pages 142–151.

Heng Ji and Ralph Grishman. 2008. Refining event extraction through cross-document inference. In *Proc. ACL*, pages 254–262.

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

Shasha Liao and Ralph Grishman. 2010a. Filtered ranking for bootstrapping in event extraction. In *Proc. COLING*, pages 680–688.

Shasha Liao and Ralph Grishman. 2010b. Using document level cross-event inference to improve event extraction. In *Proc. ACL*, pages 789–797.

Ellen Riloff. 1996. Automatically generating extraction patterns from untagged text. In *Proc. AAAI*, pages 1044–1049.

Dan Roth, Mark Sammons, and V. G. Vinod Vydiswaran. 2009. A framework for entailed relation recognition. In *Proc. ACL-IJCNLP Short Papers*, pages 57–60.

Satoshi Sekine and Akira Oda. 2007. System demonstration of on-demand information extraction. In *Proc. ACL Demo and Poster Sessions*, pages 17–20.

Satoshi Sekine. 2006. On-demand information extraction. In *Proc. COLING-ACL Poster Sessions*, pages 731–738.

Yusuke Shinyama and Satoshi Sekine. 2006. Preemptive information extraction using unrestricted relation discovery. In *Proc. NAACL*, pages 304–311.

Mark Stevenson and Mark A. Greenwood. 2005. A semantic approach to IE pattern induction. In *Proc. ACL*, pages 379–386.

Roman Yangarber, Ralph Grishman, Pasi Tapanainen, and Silja Huttunen. 2000. Automatic acquisition of domain knowledge for information extraction. In *Proc. COLING*, pages 940–946.