InToEventS: An Interactive Toolkit for Discovering and Building Event Schemas

Germán Ferrero
Universidad Nacional de Córdoba
ferrero.gf@gmail.com

Audi Primadhanty
Universitat Politècnica
de Catalunya
primadhanty@cs.upc.edu

Ariadna Quattoni
Xerox Research Centre Europe
ariadna.quattoni@
xrce.xerox.com

Abstract

Event Schema Induction is the task of learning a representation of events (e.g., bombing) and the roles involved in them (e.g, victim and perpetrator). This paper presents InToEventS, an interactive tool for learning these schemas. InToEventS allows users to explore a corpus and discover which kind of events are present. We show how users can create useful event schemas using two interactive clustering steps.

1 Introduction

An event schema is a structured representation of an event, it defines a set of atomic predicates or facts and a set of role slots that correspond to the typical entities that participate in the event. For example, a bombing event schema could consist of atomic predicates (e.g., *detonate, blow up, plant, explode, defuse* and *destroy*) and role slots for a perpetrator (the person who detonates plants or blows up), instrument (the object that is planted, detonated or defused) and a target (the object that is destroyed or blown up). Event schema induction is the task of inducing event schemas from a textual corpus. Once the event schemas are defined, slot filling is the task of extracting the instances of the events and their corresponding participants from a document.

In contrast with information extraction systems that are based on atomic relations, event schemas allow for a richer representation of the semantics of a particular domain. But, while there has been a significant amount of work in relation discovery, the task of unsupervised event schema induction has received less attention. Some unsupervised approaches have been proposed (Chambers and Jurafsky, 2011; Cheung et al., 2013; Chambers, 2013; Nguyen et al., 2015). However, they all end up assuming some form of supervision at document level, and the task of inducing event schemas in a scenario where there is no annotated data is still an open problem. This is probably because without some form of supervision we do not even have a clear way of evaluating the quality of the induced event schemas.

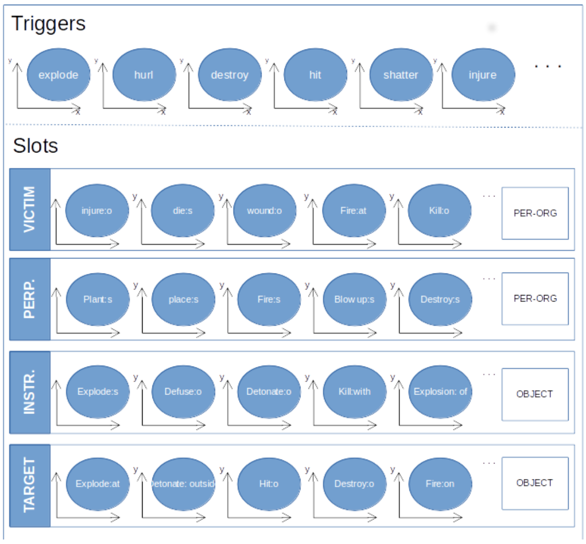


Figure 1: Event Schema Definition

In this paper we take a different approach. We argue that there is a need for an interactive event schema induction system. The tool we present enables users to explore a corpus while discovering and defining the set of event schemas that best describes the domain. We believe that such a tool addresses a realistic scenario in which the user does not know in advance the event schemas that he is interested in and he needs to explore the corpus to better define his information needs.

The main contribution of our work is to present an interactive event schema induction system that can be used by non-experts to explore a corpus and

Proceedings of the EACL 2017 Software Demonstrations, Valencia, Spain, April 3-7 2017, pages 104–107
©2017 Association for Computational Linguistics

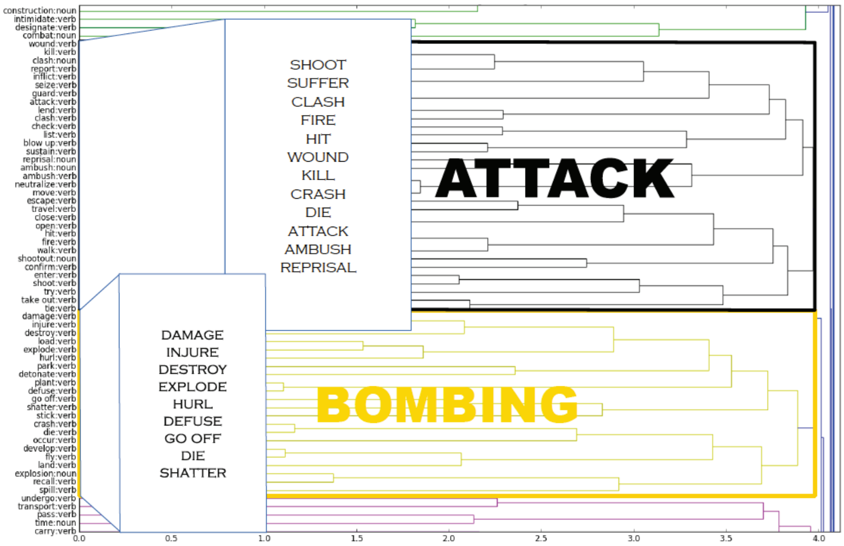


Figure 2: Event Clustering

easily build event schemas and their corresponding extractors.

The paper is organized as follows. Section 2 describes our event schema representation. Section 3 describes the interactive process for event schema discovery. Section 4 gives a short overview of related work. Finally section 5 concludes.

2 Event Schema

Our definition of event schema follows closely that of Chambers and Jurafsky (2011). An event schema consists of two main components:

Event Triggers. These correspond to a set of atomic predicates associated with an event. For example, as shown in Figure 1, for the *bombing* event schema we might have the atomic predicates: *explode, hurl, destroy, hit, shatter* and *injure*. Each atomic predicate is represented by a tuple composed of a literal (e.g. *explode*), a real-valued word vector representation and a distance threshold that defines a ball around the literal in a word vector space representation.

Event Slots. These correspond to the set of participating entities involved in the event. For example, for the *bombing* event schema we might have the event slots: *victim, perpetrator, instrument* and *target*. Each slot is represented by a tuple consisting of an entity type (e.g. person, organization or object) and a set of predicates, for example for the *victim* slot, the predicates are *injured, dies, wounded, fired* and *killed*. Each predicate is in turn represented by a tuple, consisting of a literal, a syntactic relation, a word vector and a distance threshold that defines a semantic ball. For example, the *injured* predicate is represented by the literal: *injure*, and the syntactic relation: object. This tuple is designed to represent the fact that a victim is a person or organization who has been injured, and whose corresponding word vector representation is inside a given semantic ball.

3 Event Schema Induction

We now describe InToEventS, an interactive system that allows a user to explore a corpus and build event schemas. Like in (Chambers and Jurafsky, 2011) the process is divided in two main steps. First, the user will discover the events present in the corpus (that is, event trigger sets) by interactively defining a soft partition of the predicate literals observed in the corpus. Depending on his information needs he will chose a subset of the clusters that correspond to the events that he is interested in representing. In the second step, for each chosen event trigger set, the user will complete the event schema and build slots or semantic roles via an interactive clustering of the syntactic arguments of the atomic predicates in the event trigger set.

3.1 First Step: Event Induction

To build event trigger sets we will cluster predicate literals observed in the corpus. To do this we first need to compute a distance between the predicate literals. Our notion of distance is based on two simple observations: (1) literals that tend to appear nearby in a document usually play a role in the same event description (e.g., This morning

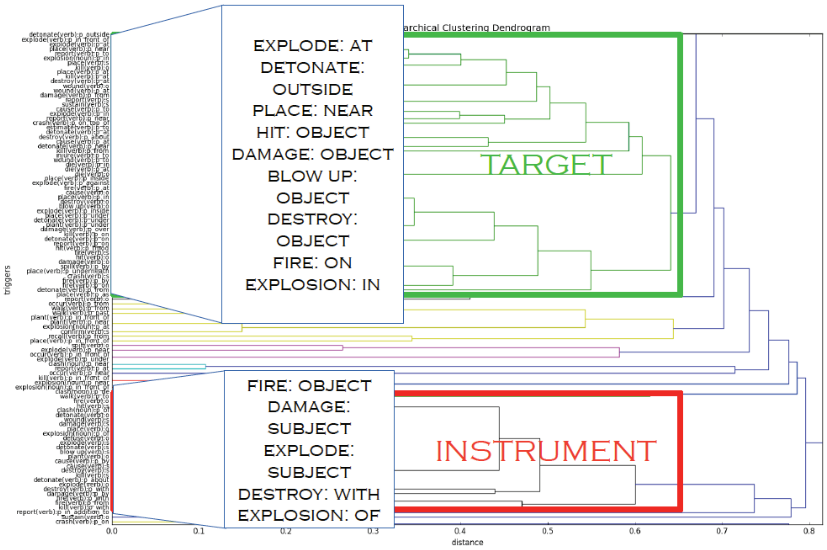


Figure 3: Slots Clustering

a terrorist *planted* a bomb, thankfully the police *defused* it before it *blew up*); and (2) literals with similar meaning are usually describing the same atomic predicates (e.g., *destroy* and *blast*).

We first extract all unique *verbs* and all nouns *noun* with a corresponding synset in Wordnet labeled as *noun.event* or *noun.act*, these are our predicate literals. Then, for each pair of literals we compute their distance taking into account the probability that they will appear nearby in a document sampled from the corpus and the distance of the corresponding literals in a word embedding vector space [1].

Once the distance is computed, we run agglomerative clustering. In the interactive step we let the user explore the resulting dendrogram and chose a distance threshold that will result in an initial partition of the predicate literals into event trigger sets (see Figure 2). In a second step, the user can merge or split the initial clusters. In a third step, the user selects and labels the clusters that he is interested in, this will become incomplete event schemas. Finally, using the word embedding representation the user has the option of expanding each event trigger set by adding predicate literals that are close in the word vector space.

3.2 Second Step: Role Induction

In this step we will complete the event schemas of the previous step with corresponding semantic roles or slots. This process is based on a simple idea: let's assume for instance a bombing event with triggers: {attack, blow up, set off, injure, die, ...}, we can intuitively describe a *victim* of a *bombing* event as *"Someone who dies, is attacked or injured"*, that is: *"PERSON: subject of die, object of attack, object of injured"*.

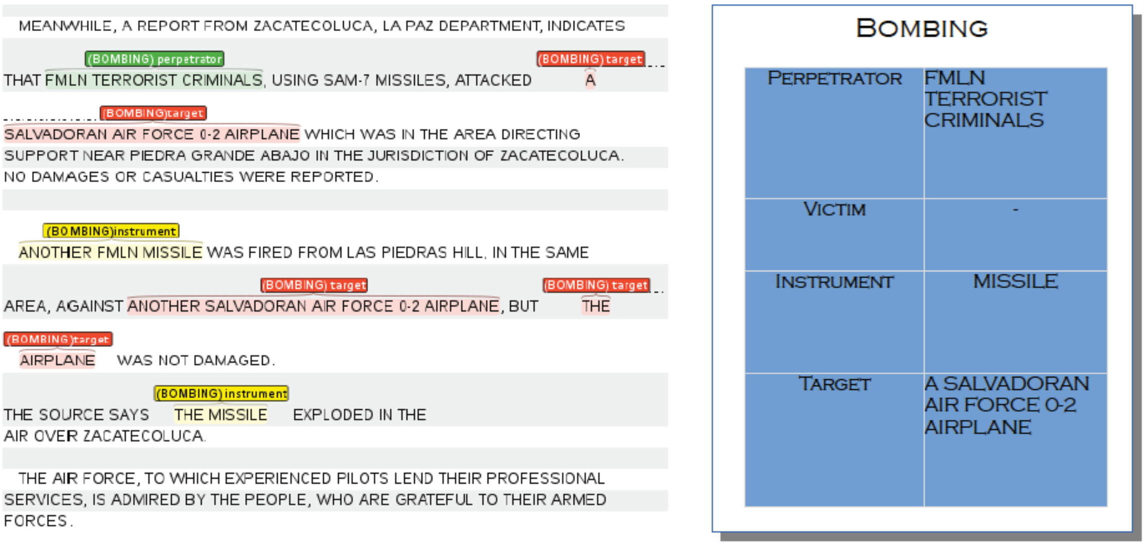
Recall that a slot is a set of predicates represented with a tuple composed by: a literal predicate and a syntactic relation, e.g. kill-subject. Additionaly each slot has an entity-type. In a first step, for each predicate in the event trigger set we extract from the corpus all unique tuples of the form predicate-syntactic relation-entity type. The extraction of such tuples uses the universal dependency representation computed by Stanford CoreNLP parser and named entity classifier.

In a second step, we compute a distance between each tuple that is based on the average word embeddings of the arguments observed in the corpus for a given tuple. For example, to compute a vector *(die, subject, PERSON)* we identify all entities of type *PERSON* in the corpus that are *subject* of the verb *die* and average their word embeddings.

Finally, as we did for event induction we run agglomerative clustering and offer the user an interactive graphic visualization of the resulting dendogram in Figure 3. The user can explore different clusters settings and store those that represent the slots that he is interested in.

Once the event schemas have been created, we can use them to annotate documents. Figure 4 shows an example of an annotated document.

[1] For experiments we used the pre-trained 300 dimensional GoogleNews model from word2vec.



MEANWHILE, A REPORT FROM ZACATECOLUCA, LA PAZ DEPARTMENT, INDICATES THAT FMLN TERRORIST CRIMINALS, USING SAM-7 MISSILES, ATTACKED A SALVADORAN AIR FORCE 0-2 AIRPLANE WHICH WAS IN THE AREA DIRECTING SUPPORT NEAR PIEDRA GRANDE ABAJO IN THE JURISDICTION OF ZACATECOLUCA. NO DAMAGES OR CASUALTIES WERE REPORTED.

ANOTHER FMLN MISSILE WAS FIRED FROM LAS PIEDRAS HILL, IN THE SAME AREA, AGAINST ANOTHER SALVADORAN AIR FORCE 0-2 AIRPLANE, BUT THE AIRPLANE WAS NOT DAMAGED.

THE SOURCE SAYS THE MISSILE EXPLODED IN THE AIR OVER ZACATECOLUCA.

THE AIR FORCE, TO WHICH EXPERIENCED PILOTS LEND THEIR PROFESSIONAL SERVICES, IS ADMIRED BY THE PEOPLE, WHO ARE GRATEFUL TO THEIR ARMED FORCES.

BOMBING	
PERPETRATOR	FMLN TERRORIST CRIMINALS
VICTIM	-
INSTRUMENT	MISSILE
TARGET	A SALVADORAN AIR FORCE 0-2 AIRPLANE

Figure 4: Slot Filling

4 Previous Work

To the best of our knowledge there is no previous work on interactive workflows for event schema induction. The most closely related work is on interactive relation extraction. Thilo and Alan (2015) presented a web toolkit for exploratory relation extraction that allows users to explore a corpus and build extraction patterns. Ralph and Yifan (2014) presented a system where users can create extractors for predifined entities and relations. Their approach is based on asking the user for seeding example instances which are then exploited with a semi-supervised learning algorithm. Marjorie et al. (2011) presented a system for interactive relation extraction based on active learning and boostrapping.

5 Conclusion

We have presented an interactive system for event schema induction, like in (Chambers and Jurafsky, 2011) the workflow is based on reducing the problem to two main clustering steps. Our system lets the user interact with the clustering process in a simple and intuitive manner and explore the corpus to create the schemas that better fits his information needs.

References

Nathanael Chambers and Dan Jurafsky. 2011. Template-based information extraction without the templates. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 976–986, Stroudsburg, PA, USA. Association for Computational Linguistics.

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

Jackie Chi Kit Cheung, Hoifung Poon, and Lucy Vanderwende. 2013. Probabilistic frame induction. *arXiv preprint arXiv:1302.4813*.

Freedman Marjorie, Ramshaw Lance, Boschee Elizabeth, Gabbard Ryan, Kratkiewicz Gary, Ward Nicolas, and Weishedel Ralph. 2011. Extreme extraction: machine reading in a week. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2011)*.

Kiem-Hieu Nguyen, Xavier Tannier, Olivier Ferret, and Romaric Besançon. 2015. Generative event schema induction with entity disambiguation. In *Proceedings of the 53rd annual meeting of the Association for Computational Linguistics (ACL-15)*.

Grishman Ralph and He Yifan. 2014. An information extraction customizer. In *Proceedings of Text, Speech and Dialogue*.

Michael Thilo and Akbik Alan. 2015. Schnapper: A web toolkit for exploratory relation extraction. In *Proceedings of ACL-IJCNLP 2015 System Demostration (ACL-15)*.