Generative Event Schema Induction with Entity Disambiguation

Kiem-Hieu Nguyen[1, 2] **Xavier Tannier**[3, 1] **Olivier Ferret**[2] **Romaric Besançon**[2]
(1) LIMSI-CNRS
(2) CEA, LIST, Laboratoire Vision et Ingnierie des Contenus, F-91191, Gif-sur-Yvette
(3) Univ. Paris-Sud
{nguyen,xtannier}@limsi.fr, {olivier.ferret,romaric.besancon}@cea.fr

Abstract

This paper presents a generative model to event schema induction. Previous methods in the literature only use head words to represent entities. However, elements other than head words contain useful information. For instance, *an armed man* is more discriminative than *man*. Our model takes into account this information and precisely represents it using probabilistic topic distributions. We illustrate that such information plays an important role in parameter estimation. Mostly, it makes topic distributions more coherent and more discriminative. Experimental results on benchmark dataset empirically confirm this enhancement.

1 Introduction

Information Extraction was initially defined (and is still defined) by the MUC evaluations (Grishman and Sundheim, 1996) and more specifically by the task of template filling. The objective of this task is to assign event roles to individual textual mentions. A template defines a specific type of events (*e.g.* earthquakes), associated with semantic roles (or slots) hold by entities (for earthquakes, their location, date, magnitude and the damages they caused (Jean-Louis et al., 2011)).

Schema induction is the task of learning these templates with no supervision from unlabeled text. We focus here on *event* schema induction and continue the trend of generative models proposed earlier for this task. The idea is to group together entities corresponding to the same role in an event template based on the similarity of the relations that these entities hold with predicates. For example, in a corpus about terrorist attacks, entities that are objects of verbs *to kill*, *to attack* can be grouped together and characterized by a role named VICTIM. The output of this identification operation is a set of clusters of which members are both words and relations, associated with their probability (see an example later in Figure 4). These clusters are not labeled but each of them represents an event slot.

Our approach here is to improve this initial idea by entity disambiguation. Some ambiguous entities, such as *man* or *soldier*, can match two different slots (victim or perpetrator). An entity such as *terrorist* can be mixed up with victims when articles relate that a terrorist has been killed by police (and thus is object of *to kill*). Our hypothesis is that the immediate context of entities is helpful for disambiguating them. For example, the fact that *man* is associated with *armed*, *dangerous*, *heroic* or *innocent* can lead to a better attribution and definition of roles. We then introduce relations between entities and their attributes in the model by means of syntactic relations.

The document level, which is generally a center notion in topic modeling, is not used in our generative model. This results in a simpler, more intuitive model, where observations are generated from slots, that are defined by probabilistic distributions on entities, predicates and syntactic attributes. This model offers room for further extensions since multiple observations on an entity can be represented in the same manner.

Model parameters are estimated by Gibbs sampling. We evaluate the performance of this approach by an automatic and empiric mapping between slots from the system and slots from the reference in a way similar to previous work in the domain.

The rest of this paper is organized as follows: Section 2 briefly presents previous work; in Section 3, we detail our entity and relation representation; we describe our generative model in Section 4, before presenting our experiments and evaluations in Section 5.

Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing, pages 188–197,
Beijing, China, July 26-31, 2015.

2 Related Work

Despite efforts made for making template filling as generic as possible, it still depends heavily on the type of events. Mixing generic processes with a restrictive number of domain-specific rules (Freedman et al., 2011) or examples (Grishman and He, 2014) is a way to reduce the amount of effort needed for adapting a system to another domain. The approaches of *On-demand information extraction* (Hasegawa et al., 2004; Sekine, 2006) and *Preemptive Information Extraction* (Shinyama and Sekine, 2006) tried to overcome this difficulty in another way by exploiting templates induced from representative documents selected by queries.

Event schema induction takes root in work on the acquisition from text of knowledge structures, such as the Memory Organization Packets (Schank, 1980), used by early text understanding systems (DeJong, 1982) and more recently by Ferret and Grau (1997). First attempts for applying such processes to schema induction have been made in the fields of Information Extraction (Collier, 1998), Automatic Summarization (Harabagiu, 2004) and event Question-Answering (Filatova et al., 2006; Filatova, 2008).

More recently, work after (Hasegawa et al., 2004) has developed weakly supervised forms of Information Extraction including schema induction in their objectives. However, they have been mainly applied to binary relation extraction in practice (Eichler et al., 2008; Rosenfeld and Feldman, 2007; Min et al., 2012). In parallel, several approaches were proposed for performing specifically schema induction in already existing frameworks: clause graph clustering (Qiu et al., 2008), event sequence alignment (Regneri et al., 2010) or LDA-based approach relying on FrameNet-like semantic frames (Bejan, 2008). More event-specific generative models were proposed by Chambers (2013) and Cheung et al. (2013). Finally, Chambers and Jurafsky (2008), Chambers and Jurafsky (2009), Chambers and Jurafsky (2011), improved by Balasubramanian et al. (2013), and Chambers (2013) focused specifically on the induction of event roles and the identification of chains of events for building representations from texts by exploiting coreference resolution or the temporal ordering of events. All this work is also linked to work about the induction of scripts from texts, more or less closely linked to events, such as (Frermann et al., 2014), (Pichotta and Mooney, 2014) or (Modi and Titov, 2014).

The work we present in this article is in line with Chambers (2013), which will be described in more details in Section 5, together with a quantitative and qualitative comparison.

	Attributes	**Head**	**Triggers**
#1	[armed:amod]	man	[attack:nsubj, kill:nsubj]
#2	[police:nn]	station	[attack:dobj]
#3	[]	policeman	[kill:dobj]
#4	[innocent:amod, young:amod]	man	[wound:dobj]

Figure 1: Entity representation as tuples of ([attributes], head, [triggers]).

3 Entity Representation

An entity is represented as a triple containing: a head word h, a list A of attribute relations and a list T of trigger relations. Consider the following example:

(1) Two armed men attacked the police station and killed a policeman. An innocent young man was also wounded.

As illustrated in Figure 1, four entities, equivalent to four separated triples, are generated from the text above. Head words are extracted from noun phrases. A trigger relation is composed of a predicate (*attack*, *kill*, *wound*) and a dependency type (subject, object). An attribute relation is composed of an argument (*armed*, *police*, *young*) and a dependency type (adjectival, nominal or verbal modifier). In the relationship to triggers, a head word is argument, but in the relationship to attributes, it is predicate. We use Stanford NLP toolkit (Manning et al., 2014) for parsing and coreference resolution.

A head word is extracted if it is a nominal or proper noun and it is related to at least one trigger; pronouns are omitted. A trigger of an head word is extracted if it is a verb or an eventive noun and the head word serves as its subject, object, or preposition. We use the categories *noun.EVENT* and *noun.ACT* in WordNet as a list of eventive nouns. A head word can have more than one trigger. These multiple relations can come from a syntactic coordination inside a single sentence, as it is the case in the first sentence of the illustrating example. They can also represent a coreference

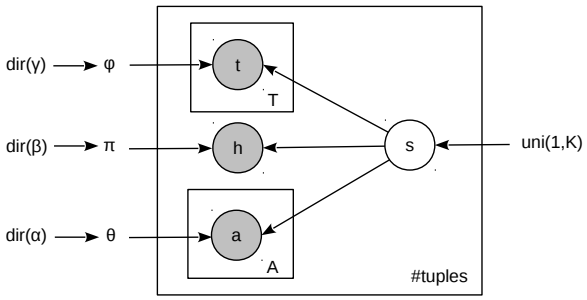


Figure 2: Generative model for event induction.

chain across sentences, as we use coreference resolution to merge the triggers of mentions coreferring to the same entity in a document. Coreferences are useful sources for event induction (Chambers and Jurafsky, 2011; Chambers, 2013). Finally, an attribute is extracted if it is an adjective, a noun or a verb and serves as an adjective, verbal or nominal modifier of a head word. If there are several modifiers, only the closest to the head word is selected. This "best selection" heuristic allows to omit non-discriminative attributes for the entity.

4 Generative Model

4.1 Model Description

Figure 2 shows the plate notation of our model. For each triple representing an entity e, the model first assigns a slot s for the entity from an uniform distribution $uni(1, K)$. Its head word h is then generated from a multinominal distribution π_s. Each t_i of event trigger relations T_e is generated from a multinominal distribution ϕ_s. Each a_j of attribute relations A_e is similarly generated from a multinominal distribution θ_s. The distributions θ, π, and ϕ are generated from Dirichlet priors $dir(\alpha)$, $dir(\beta)$ and $dir(\gamma)$ respectively.

Given a set of entities E, our model (π, ϕ, θ) is defined by

$$P_{\pi,\phi,\theta}(E) = \prod_{e \in E} P_{\pi,\phi,\theta}(e) \quad (2)$$

where the probability of each entity e is defined by

$$\begin{aligned} P_{\pi,\phi,\theta}(e) &= P(s) \\ &\times P(h|s) \\ &\times \prod_{t \in T_e} P(t|s) \\ &\times \prod_{a \in A_e} P(a|s) \end{aligned} \quad (3)$$

The generative story is as follows:

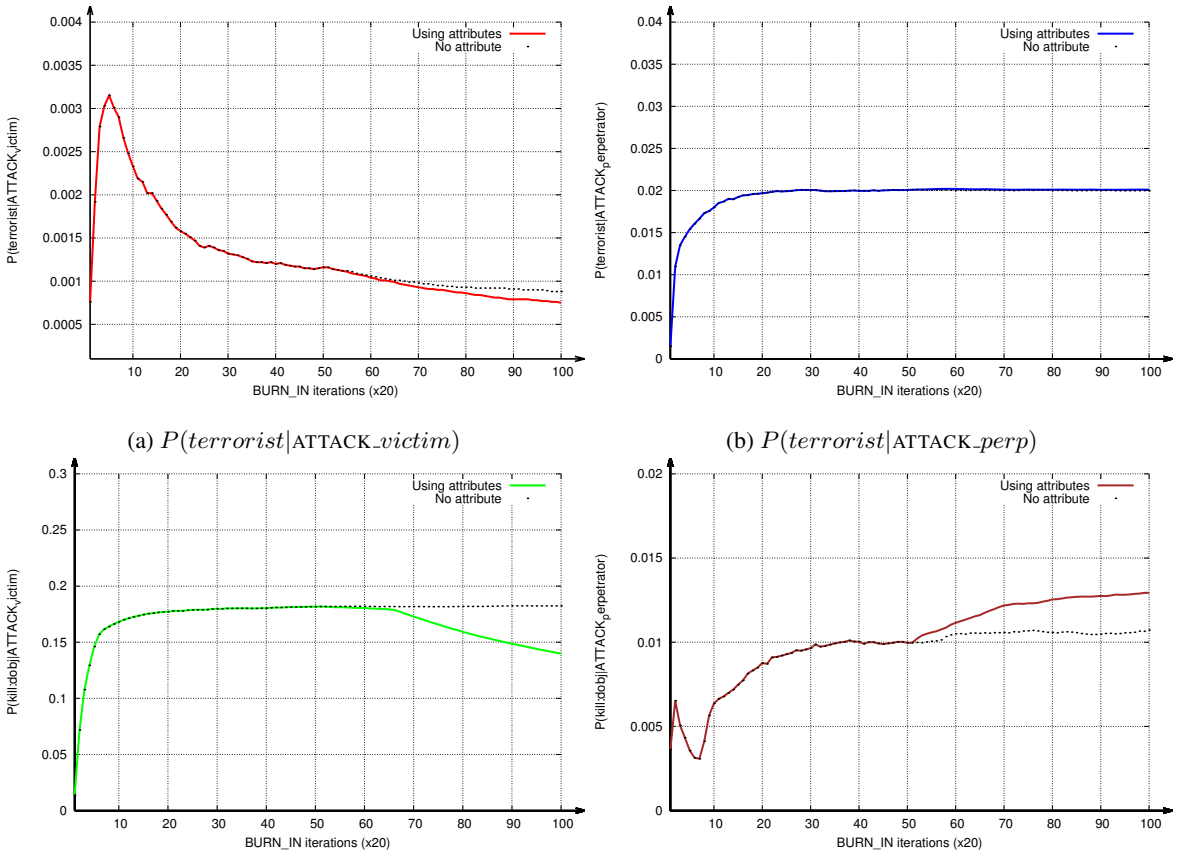
for *slot* $s \leftarrow 1$ **to** K **do**
  Generate an attribute distribution θ_s from a Dirichlet prior $dir(\alpha)$;
  Generate a head distribution π_s from a Dirichlet prior $dir(\beta)$;
  Generate a trigger distribution ϕ_s from a Dirichlet prior $dir(\gamma)$;
end
for *entity* $e \in E$ **do**
  Generate a slot s from a uniform distribution $uni(1, K)$;
  Generate a head h from a multinominal distribution π_s;
  for $i \leftarrow 1$ **to** $|T_e|$ **do**
    Generate a trigger t_i from a multinominal distribution ϕ_s;
  end
  for $j \leftarrow 1$ **to** $|A_e|$ **do**
    Generate an attribute a_j from a multinominal distribution ϕ_s;
  end
end

4.2 Parameter Estimation

For parameter estimation, we use the Gibbs sampling method (Griffiths, 2002). The slot variable s is sampled by integrating out all the other variables.

Previous models (Cheung et al., 2013; Chambers, 2013) are based on document-level topic modeling, which originated from models such as Latent Dirichlet Allocation (Blei et al., 2003). Our model is, instead, independent from document contexts. Its input is a sequence of entity triples. Document boundary is only used in a post-processing step of filtering (see Section 5.3 for more details). There is a universal slot distribution instead of each slot distribution for one document. Furthermore, slot prior is ignored by using a uniform distribution as a particular case of categorical probability. Sampling-based slot assignment could depend on initial states and random seeds. In our implementation of Gibbs sampling, we use 2,000 *burn-in* of overall 10,000 iterations. The purpose of *burn-in* is to assure that parameters converge to a stable state before estimating the probability distributions. Moreover, an interval step of 100 is applied between consecutive samples in order to avoid too strong coherence.

Particularly, for tracking changes in probabilities resulting from attribute relations, we ran in the first stage a specific burn-in with only heads and trigger relations. This stable state was then used as initialization for the second burn-in in



(a) $P(terrorist|\text{ATTACK\_}victim)$

(b) $P(terrorist|\text{ATTACK\_}perp)$

(c) $P(kill:dobj|\text{ATTACK\_}victim)$

(d) $P(kill:dobj|\text{ATTACK\_}perp)$

Figure 3: Probability convergence when using attributes in sampling. The use of attributes is started at point 50 (*i.e.*, 50% of burn-in phase). The dotted line shows convergence without attributes; the continuous line shows convergence with attributes.

which attributes, heads, and triggers were used altogether. This specific experimental setting made us understand how the attributes modified distributions. We observed that non-ambiguous words or relations (*i.e. explode*, *murder:nsubj*) were only slightly modified whereas probabilities of ambiguous words such as *man*, *soldier* or triggers such as *kill:dobj* or *attack:nsubj* converged smoothly to a different stable state that was semantically more coherent. For instance, the model interestingly realized that even if a *terrorist* was killed (*e.g.* by police), he was not actually a real victim of an attack. Figure 3 shows probability convergences of *terrorist* and *kill:dobj* given ATTACK_victim and ATTACK_perpetrator.

5 Evaluations

In order to compare with related work, we evaluated our method on the Message Understanding Conference (MUC-4) corpus (Sundheim, 1991) using precision, recall and F-score as conventional metrics for template extraction.

In what follows, we first introduce the MUC-4 corpus (Section 5.1.1), we detail the mapping technique between learned slots and reference slots (5.1.2) as well as the hyper-parameters of our model (5.1.3). Next, we present a first experiment (Section 5.2) showing how using attribute relations improves overall results. The second experiment (Section 5.3) studies the impact of document classification. We then compare our results with previous approaches, more particularly with Chambers (2013), from both quantitative and qualitative points of view (Section 5.4). Finally, Section 5.5 is dedicated to error analysis, with a special emphasis on sources of false positives.

5.1 Experimental Setups

5.1.1 Datasets

The MUC-4 corpus contains 1,700 news articles about terrorist incidents happening in Latin America. The corpus is divided into 1,300 documents

for the development set and four test sets, each containing 100 documents.

We follow the rules in the literature to guarantee comparable results (Patwardhan and Riloff, 2007; Chambers and Jurafsky, 2011). The evaluation focuses on four template types – ARSON, ATTACK, BOMBING, KIDNAPPING – and four slots – Perpetrator, Instrument, Target, and Victim. Perpetrator is merged from Perpetrator_Individual and Perpetrator_Organization. The matching between system answers and references is based on head word matching. A head word is defined as the right-most word of the phrase or as the right-most word of the first 'of' if the phrase contains any. Optional templates and slots are ignored when calculating recall. Template types are ignored in evaluation: this means that a perpetrator of BOMBING in the answers could be compared to a perpetrator of ARSON, ATTACK, BOMBING or KIDNAPPING in the reference.

5.1.2 Slot Mapping

The model learns K slots and assigns each entity in a document to one of the learned slots. Slot mapping consists in matching each reference slot to an equivalent learned slot.

Note that among the K learned slots, some are irrelevant while others, sometimes of high quality, contain entities that are not part of the reference (spatio-temporal information, protagonist context, etc.). For this reason, it makes sense to have much more learned slots than expected event slots.

Similarly to previous work in the literature, we implemented an automatic empirical-driven slot mapping. Each reference slot was mapped to the learned slot that performed the best on the task of template extraction according to the F-score metric. Here, two identical slots of two different templates, such as ATTACK_victim and KIDNAPPING_victim, must to be mapped separately. Figure 4 shows the most common words of two learned slots which were mapped to BOMBING_instrument and KIDNAPPING_victim. This mapping is then kept for testing.

5.1.3 Parameter Tuning

We first tuned hyper-parameters of the models on the development set. The number of slots was set to $K = 35$. Dirichlet priors were set to $\alpha = 0.1$, $\beta = 1$ and $\gamma = 0.1$. The model was learned from the whole dataset. Slot mapping was done on tst1 and tst2. Outputs from tst3 and tst4 were evaluated using references and were averaged across ten runs.

BOMBING_instrument

Attributes	Heads	Triggers
car:nn	bomb	explode:nsubj
powerful:amod	fire	hear:dobj
explosive:amod	explosion	place:dobj
dynamite:nn	blow	cause:nsubj
heavy:amod	charge	set:dobj

KIDNAPPING_victim

Attributes	Heads	Triggers
several:amod	people	arrest:dobj
other:amod	person	kidnap:dobj
responsible:amod	man	release:dobj
military:amod	member	kill:dobj
young:amod	leader	identify:prep_as

Figure 4: Attribute, head and trigger distributions learned by the model *HT+A* for learned slots that were mapped to BOMBING_instrument and KIDNAPPING_victim.

5.2 Experiment 1: Using Entity Attributes

In this experiment, two versions of our model are compared: *HT+A* uses entity heads, event trigger relations and entity attribute relations. *HT* uses only entity heads and event triggers and omits attributes.

We studied the gain brought by attribute relations with a focus on their effect when coreference information was available or was missing. The variations on the model input are named *single*, *multi* and *coref*. *Single* input has only one event trigger for each entity. A text like *an armed man attacked the police station and killed a policeman* results in two triples for the entity *man*: *(armed:amod, man, attack:nsubj)* and *(armed:amod, man, kill:nsubj)*. In *multi* input, one entity can have several event triggers, leading for the text above to the triple *(armed:amod, man, [attack:nsubj, kill:nsubj])*. The *coref* input is richer than *multi* in that, in addition to triggers from the same sentence, triggers linked to the same coreferred entity are merged together. For instance, if *man* in the above example corefers with *he* in *He was arrested three hours later*, the merged triple becomes *(armed:amod, man, [attack:nsubj, kill:nsubj, arrest:dobj])*. The plate notations of these model+data combinations are given in Figure 5.

Table 1 shows a consistent improvement when using attributes, both with and without coreferences. The best performance of 40.62 F-score is obtained by the full model on inputs with coref-

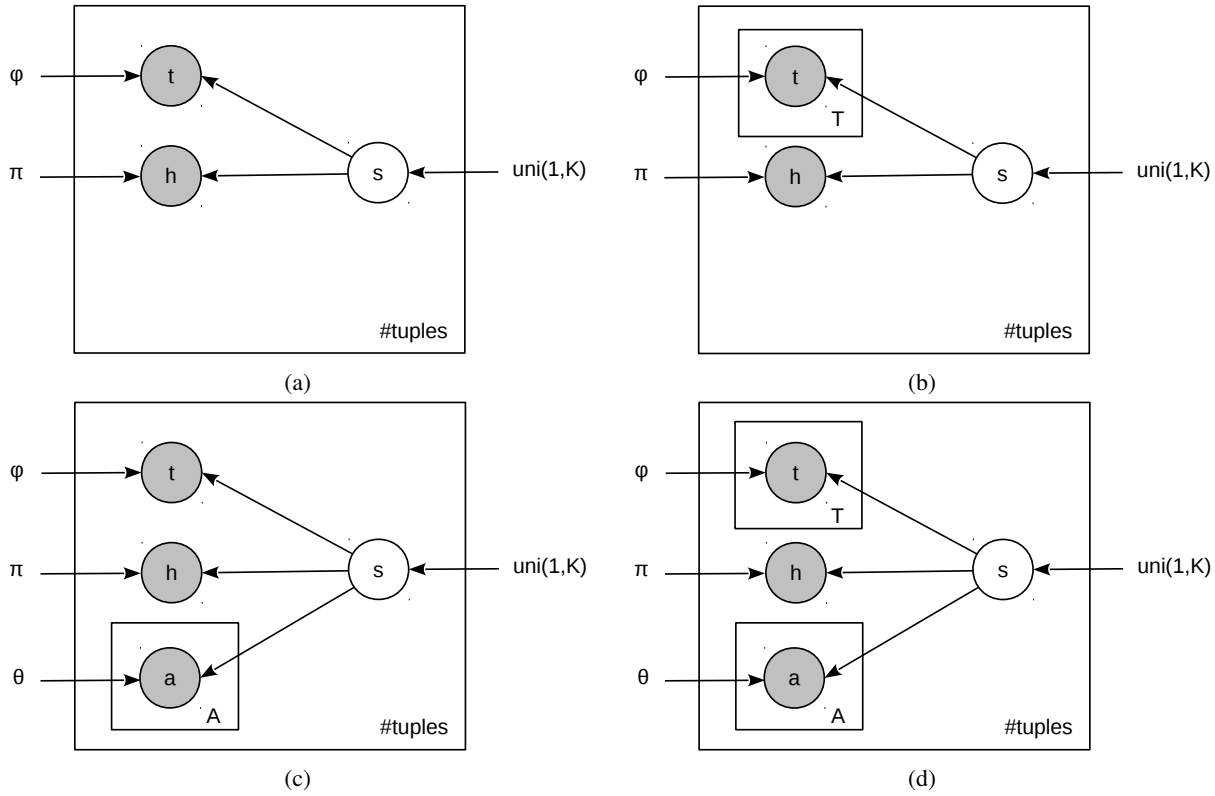


Figure 5: Model variants (Dirichlet priors are omitted for simplicity): 5a) HT model ran on single data. This model is equivalent to 5b) with T=1; 5b) HT model ran on multi data; 5c) HT+A model ran on single data; 5d) HT+A model ran on multi data.

Data	HT			HT+A		
	P	R	F	P	R	F
Single	29.59	51.17	37.48	30.22	52.41	38.33
Multi	29.32	52.21	37.52	30.82	51.68	38.55
Coref	39.99	53.53	40.01	32.42	54.59	40.62

Table 1: Improvement from using attributes.

erences. Using both attributes in the model and coreference to generate input data results in a gain of 3 F-score points.

5.3 Experiment 2: Document Classification

In the second experiment, we evaluated our model with a post-processing step of document classification.

The MUC-4 corpus contains many "irrelevant" documents. A document is irrelevant if it contains no template. Among 1,300 documents in the development set, 567 are irrelevant. The most challenging part is that there are many terrorist entities, *e.g. bomb*, *force*, *guerrilla*, occurring in irrelevant documents. That makes filtering out those documents important, but difficult. As document classification is not explicitly performed by our model, a post-processing step is needed. Document classification is expected to reduce false positives in irrelevant documents while not dramatically reducing recall.

Given a document d with slot-assigned entities and a set of mapped slots S_m resulting from slot mapping, we have to decide whether this document is relevant or not. We define the relevance score of a document as:

$$relevance(d) = \frac{\sum_{e \in d: s_e \in S_m} \sum_{t \in T_e} P(t|s_e)}{\sum_{e \in d} \sum_{t \in T_e} P(t|s_e)} \quad (4)$$

where e is an entity in the document d; s_e is the slot value assigned to e; and t is an event trigger in the list of triggers T_e.

The equation (4) defines the score of an entity as the sum of the conditional probabilities of triggers given a slot. The relevance score of the document is proportional to the score of the entities assigned to mapped slots. If this relevance score is higher than a threshold λ, then the document is considered as relevant. The value of $\lambda = 0.02$ was tuned

System	P	R	F
HT+A	32.42	54.59	40.62
HT+A + doc. classification	35.57	53.89	42.79
HT+A + oracle classification	44.58	54.59	49.08

Table 2: Improvement from document classification as post-processing.

System	P	R	F
Cheung et al. (2013)	32	37	34
Chambers and Jurafsky (2011)	**48**	25	33
Chambers (2013) (paper values)	41	41	41
HT+A + doc. classification	36	**54**	**43**

Table 3: Comparison to state-of-the-art unsupervised systems.

on the development set by maximizing the F-score of document classification.

Table 2 shows the improvement when applying document classification. The precision increases as false positives from irrelevant documents are filtered out. The loss of recall comes from relevant documents that are mistakenly filtered out. However, this loss is not significant and the overall F-score finally increases by 5%. We also compare our results to an "oracle" classifier that would remove all irrelevant documents while preserving all relevant ones. The performance of this oracle classification shows that there are some room for further improvement from document classification.

Irrelevant document filtering is a technique applied by most supervised and unsupervised approaches. Supervised methods prefer relevance detection at sentence or phrase-level (Patwardhan and Riloff, 2009; Patwardhan and Riloff, 2007). As for several unsupervised methods, Chambers (2013) includes document classification in his topic model. Chambers and Jurafsky (2011) and Cheung et al. (2013) use the learned clusters to classify documents by estimating the relevance of a document with respect to a template from *post-hoc* statistics about event triggers.

5.4 Comparison to State-of-the-Art

For comparing in more depth our results to the state-of-the-art in the literature. we reimplemented the method proposed in Chambers (2013) and integrated our attribute distributions into his model (as shown in Figure 6).

The main differences between this model and ours are the following:

1. The *full template model* of Chambers (2013) adds a distribution ψ linking events to documents. This makes the model more complex and maybe less intuitive since there is no reason to connect documents and slots (a document may contain references to several templates and slot mapping does not depend on document level). A benefit of this document distribution is that it leads to a free classification of irrelevant documents, thus avoiding a pre- or post-processing for classification. However, this issue of document relevance is very specific to the MUC corpus and the evaluation method; In a more general use case, there would be no "irrelevant" documents, only documents on various topics.

2. Each entity is linked to an event variable e. This event generates a predicate for each entity mention (recall that mentions of an entity are all occurrences of this entity in the documents, for example in a coreference chain). Our work instead focus on the fact that a probabilistic model could have multiple observations at the same position. Multiple triggers and multiple attributes are treated equally. The sources of multiple attributes and multiple triggers are not only from document-level coreferences but also from dependency relations (or even from domain-level entity coreferences if available). Hence, our model arguably generalizes better in terms of both modeling and input data.

3. Chambers (2013) applies a heuristic constraint during the sampling process, imposing that subject and object of the same predicate (*e.g. kill:nsubj* and *kill:dobj*) are not distributed into the same slot. Our model does not require this heuristic.

Some details concerning data preprocessing and model parameters are not fully specified by Chambers (2013); for this reason, our implementation of the model (applied on the same data) leads to slightly different results than those published. That is why we present the two results here (paper values in Table 3, reimplementation values in Table 4).

Table 3 shows that our model outperforms the others on recall by a large margin. It achieves the

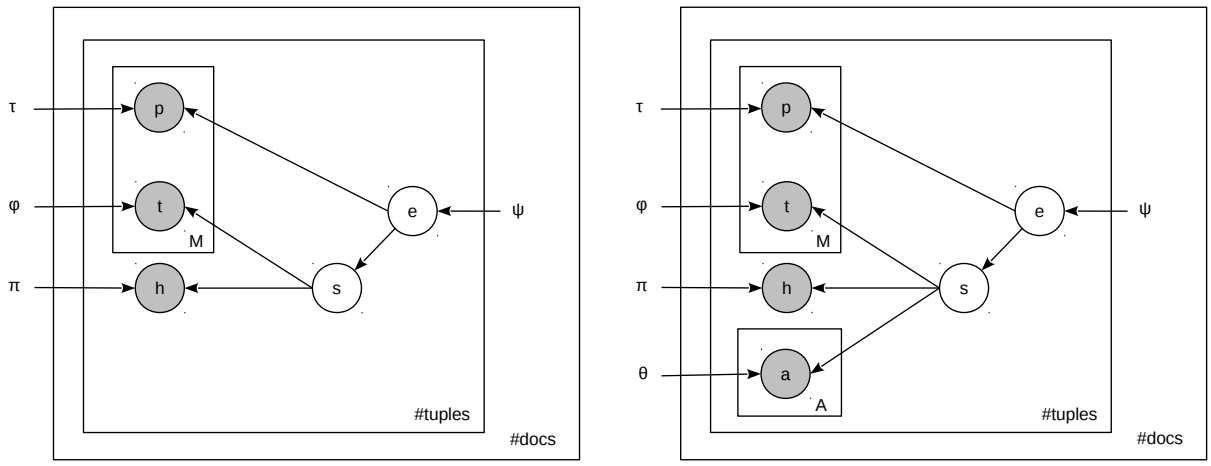


Figure 6: Variation of Chambers (2013) model: 6a) Original model; 6b) Original model + attribute distributions.

Chambers (2013)	P	R	F
Original reimpl.	38.65	42.68	40.56
Original reimpl. + Attribute	39.25	43.68	41.31

Table 4: Performance on reimplementation of Chambers (2013).

best overall F-score. In addition, as stated by our experiments, precision could be further improved by more sophisticated document classification. Interestingly, using attributes also proves to be useful in the model proposed by Chambers (2013) (as shown in Table 4).

5.5 Error Analysis

We performed an error analysis on the output of *HT+A + doc. classification* to detect the origin of false positives (FPs). 38% of FPs are mentions that never occur in the reference. Within this 38%, *attacker* and *killer* are among the most frequent errors. These words could refer to a perpetrator of an attack. These mentions, however, do not occur in the reference, possibly because human annotators consider them as too generic terms. Apart from such generic terms, other assignments are obvious errors of the system, *e.g. window*, *door* or *wall* as physical target; *action* or *massacre* as perpetrator; *explosion* or *shooting* as instrument. These kinds of errors are due to the fact that in our model, as in the one of Chambers (2013), the number of slots is fixed and is not equivalent to the real number of reference slots.

On the other hand, 62% of FPs are mentions of entities that occur at least once in the reference. On top of the list are perpetrators such as *guerrilla*, *group* and *rebel*. The model is capable of assigning *guerrilla* to *attribution* slot if it is accompanied by a trigger like *announce:nsubj*. However, triggers that describe quasi-terrorism events (*e.g.* menace, threatening, military conflict) are also grouped into *perpetrator* slots. Similarly, mentions of frequent words such as *bomb* (*instrument*), *building*, *house*, *office* (*targets*) tend to be systematically grouped into these slots, regardless of their relations. Increasing the number of slots (to sharpen their content) does not help overall. This is due to the fact that the MUC corpus is very small and is biased towards terrorism events. Adding a higher level of template type as in Chambers (2013) partially solves the problem but makes recall decrease (as shown in Table 3).

6 Conclusions and Perspectives

We presented a generative model for representing the roles played by the entities in an event template. We focused on using immediate contexts of entities and proposed a simpler and more effective model than those proposed in previous work. We evaluated this model on the MUC-4 corpus.

Even if our results outperform other unsupervised approaches, we are still far from results obtained by supervised systems. Improvements can be obtained by several ways. First, the characteristics of the MUC-4 corpus are a limiting factor. The corpus is small and roles are similar from a template to another, which does not reflect reality.

A bigger corpus, even partially annotated but presenting a better variety of templates, could lead to very different approaches.

As we showed, our model comes with a unified representation of all types of relations. This opens the way to the use of multiple types of relations (syntactic, semantic, thematic, etc.) to refine the clusters.

Last but not least, the evaluation protocol, that became a kind of *de facto* standard, is very much imperfect. Most notably, the way of finally mapping with reference slots can have a great influence on the results.

Acknowledgment

This work was partially financed by the Foundation for Scientific Cooperation "Campus Paris-Saclay" (FSC) under the project Digiteo ASTRE No. 2013-0774D.

References

Niranjan Balasubramanian, Stephen Soderland, Mausam, and Oren Etzioni. 2013. Generating Coherent Event Schemas at Scale. In *2013 Conference on Empirical Methods in Natural Language Processing (EMNLP 2013)*, pages 1721–1731, Seattle, Washington, USA, October.

Cosmin Adrian Bejan. 2008. Unsupervised Discovery of Event Scenarios from Texts. In *Twenty-First International Florida Artificial Intelligence Research Society Conference (FLAIRS 2008)*, pages 124–129, Coconut Grove, Florida.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3:993–1022, March.

Nathanael Chambers and Dan Jurafsky. 2008. Unsupervised Learning of Narrative Event Chains. In *ACL-08: HLT*, pages 789–797, Columbus, Ohio, June.

Nathanael Chambers and Dan Jurafsky. 2009. Unsupervised Learning of Narrative Schemas and their Participants. In *Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP (ACL-IJCNLP'09)*, pages 602–610, Suntec, Singapore, August.

Nathanael Chambers and Dan Jurafsky. 2011. Template-Based Information Extraction without the Templates. In *49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL 2011)*, pages 976–986, Portland, Oregon, USA, June.

Nathanael Chambers. 2013. Event Schema Induction with a Probabilistic Entity-Driven Model. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807, Seattle, Washington, USA, October.

Kit Jackie Chi Cheung, Hoifung Poon, and Lucy Vanderwende. 2013. Probabilistic Frame Induction. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 837–846.

R. Collier. 1998. *Automatic Template Creation for Information Extraction*. Ph.D. thesis, University of Sheffield.

Gerald DeJong. 1982. An overview of the FRUMP system. In W. Lehnert and M. Ringle, editors, *Strategies for natural language processing*, pages 149–176. Lawrence Erlbaum Associates.

Kathrin Eichler, Holmer Hemsen, and Günter Neumann. 2008. Unsupervised Relation Extraction From Web Documents. In *6^{th} Conference on Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco.

Olivier Ferret and Brigitte Grau. 1997. An Aggregation Procedure for Building Episodic Memory. In *15^{th} International Joint Conference on Artificial Intelligence (IJCAI-97)*, pages 280–285, Nagoya, Japan.

Elena Filatova, Vasileios Hatzivassiloglou, and Kathleen McKeown. 2006. Automatic Creation of Domain Templates. In *21^{st} International Conference on Computational Linguistics and 44^{th} Annual Meeting of the Association for Computational Linguistics (COLING-ACL 2006)*, pages 207–214, Sydney, Australia.

Elena Filatova. 2008. *Unsupervised Relation Learning for Event-Focused Question-Answering and Domain Modelling*. Ph.D. thesis, Columbia University.

Marjorie Freedman, Lance Ramshaw, Elizabeth Boschee, Ryan Gabbard, Gary Kratkiewicz, Nicolas Ward, and Ralph Weischedel. 2011. Extreme Extraction – Machine Reading in a Week. In *2011 Conference on Empirical Methods in Natural Language Processing (EMNLP 2011)*, pages 1437–1446, Edinburgh, Scotland, UK., July.

Lea Frermann, Ivan Titov, and Manfred Pinkal. 2014. A Hierarchical Bayesian Model for Unsupervised Induction of Script Knowledge. In *14th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2014)*, pages 49–57, Gothenburg, Sweden, April.

Tom Griffiths. 2002. Gibbs sampling in the generative model of Latent Dirichlet Allocation. Technical report, Stanford University.

Ralph Grishman and Yifan He. 2014. An Information Extraction Customizer. In Petr Sojka, Ale Hork, Ivan Kopeek, and Karel Pala, editors, *17th International Conference on Text, Speech and Dialogue (TSD 2014)*, volume 8655 of *Lecture Notes in Computer Science*, pages 3–10. Springer International Publishing.

Ralph Grishman and Beth Sundheim. 1996. Message Understanding Conference-6: A Brief History. In *16^{th} International Conference on Computational linguistics (COLING'96)*, pages 466–471, Copenhagen, Denmark.

Sanda Harabagiu. 2004. Incremental Topic Representation. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING'04)*, Geneva, Switzerland, August.

Takaaki Hasegawa, Satoshi Sekine, and Ralph Grishman. 2004. Discovering Relations among Named Entities from Large Corpora. In *42^{nd} Meeting of the Association for Computational Linguistics (ACL'04)*, pages 415–422, Barcelona, Spain.

Ludovic Jean-Louis, Romaric Besanon, and Olivier Ferret. 2011. Text Segmentation and Graph-based Method for Template Filling in Information Extraction. In *5^{th} International Joint Conference on Natural Language Processing (IJCNLP 2011)*, pages 723–731, Chiang Mai, Thailand.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP Natural Language Processing Toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60, Baltimore, USA, jun.

Bonan Min, Shuming Shi, Ralph Grishman, and Chin-Yew Lin. 2012. Ensemble Semantics for Large-scale Unsupervised Relation Extraction. In *2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL 2012*, pages 1027–1037, Jeju Island, Korea.

Ashutosh Modi and Ivan Titov. 2014. Inducing neural models of script knowledge. In *Eighteenth Conference on Computational Natural Language Learning (CoNLL 2014)*, pages 49–57, Ann Arbor, Michigan.

Siddharth Patwardhan and Ellen Riloff. 2007. Effective Information Extraction with Semantic Affinity Patterns and Relevant Regions. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL 2007)*, pages 717–727, Prague, Czech Republic, June.

Siddharth Patwardhan and Ellen Riloff. 2009. A Unified Model of Phrasal and Sentential Evidence for Information Extraction. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing (EMNLP 2009)*, pages 151–160.

Karl Pichotta and Raymond Mooney. 2014. Statistical script learning with multi-argument events. In *14th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2014)*, pages 220–229, Gothenburg, Sweden.

Long Qiu, Min-Yen Kan, and Tat-Seng Chua. 2008. Modeling Context in Scenario Template Creation. In *Third International Joint Conference on Natural Language Processing (IJCNLP 2008)*, pages 157–164, Hyderabad, India.

Michaela Regneri, Alexander Koller, and Manfred Pinkal. 2010. Learning Script Knowledge with Web Experiments. In *48th Annual Meeting of the Association for Computational Linguistics (ACL 2010)*, pages 979–988, Uppsala, Sweden, July.

Benjamin Rosenfeld and Ronen Feldman. 2007. Clustering for unsupervised relation identification. In *Sixteenth ACM conference on Conference on information and knowledge management (CIKM'07)*, pages 411–418, Lisbon, Portugal.

Roger C. Schank. 1980. Language and memory. *Cognitive Science*, 4:243–284.

Satoshi Sekine. 2006. On-demand information extraction. In *21^{st} International Conference on Computational Linguistics and 44^{th} Annual Meeting of the Association for Computational Linguistics (COLING-ACL 2006)*, pages 731–738, Sydney, Australia.

Yusuke Shinyama and Satoshi Sekine. 2006. Preemptive Information Extraction using Unrestricted Relation Discovery. In *HLT-NAACL 2006*, pages 304–311, New York City, USA.

Beth M. Sundheim. 1991. Third Message Understanding Evaluation and Conference (MUC-3): Phase 1 Status Report. In *Proceedings of the Workshop on Speech and Natural Language*, HLT '91, pages 301–305.