Using Document Level Cross-Event Inference to Improve Event Extraction

Shasha Liao
New York University
715 Broadway, 7th floor
New York, NY 10003 USA
liaoss@cs.nyu.edu

Ralph Grishman
New York University
715 Broadway, 7th floor
New York, NY 10003 USA
grishman@cs.nyu.edu

Abstract

Event extraction is a particularly challenging type of information extraction (IE). Most current event extraction systems rely on local information at the phrase or sentence level. However, this local context may be insufficient to resolve ambiguities in identifying particular types of events; information from a wider scope can serve to resolve some of these ambiguities. In this paper, we use document level information to improve the performance of ACE event extraction. In contrast to previous work, we do not limit ourselves to information about events of the same type, but rather use information about other types of events to make predictions or resolve ambiguities regarding a given event. We learn such relationships from the training corpus and use them to help predict the occurrence of events and event arguments in a text. Experiments show that we can get 9.0% (absolute) gain in trigger (event) classification, and more than 8% gain for argument (role) classification in ACE event extraction.

1 Introduction

The goal of event extraction is to identify instances of a class of events in text. The ACE 2005 event extraction task involved a set of 33 generic event types and subtypes appearing frequently in the news. In addition to identifying the event itself, it also identifies all of the *participants* and *attributes* of each event; these are the *entities* that are involved in that event.

Identifying an event and its participants and attributes is quite difficult because a larger field of view is often needed to understand how facts tie together. Sometimes it is difficult even for people to classify events from isolated sentences. From the sentence:

*(1) He **left** the company.*

it is hard to tell whether it is a *Transport* event in ACE, which means that he left the place; or an *End-Position* event, which means that he retired from the company.

However, if we read the whole document, a clue like "*he planned to go shopping before he went home*" would give us confidence to tag it as a *Transport* event, while a clue like "*They held a party for his retirement*" would lead us to tag it as an *End-Position* event.

Such clues are evidence from the same event type. However, sometimes another event type is also a good predictor. For example, if we find a *Start-Position* event like "*he was named president three years ago*", we are also confident to tag (1) as *End-Position* event.

Event argument identification also shares this benefit. Consider the following two sentences:

*(2) A bomb **exploded** in Bagdad; seven people **died** while 11 were **injured**.*

*(3) A bomb **exploded** in Bagdad; the suspect got **caught** when he tried to escape.*

If we only consider the local context of the trigger "*exploded*", it is hard to determine that "*seven people*" is a likely *Target* of the *Attack* event in (2), or that the "*suspect*" is the *Attacker* of the *Attack* event, because the structures of (2) and (3) are quite similar. The only clue is from the semantic inference that a person who died may well have been a *Target* of the *Attack* event, and the person arrested is probably the *Attacker* of the *Attack* event. These may be seen as

Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, pages 789–797,
Uppsala, Sweden, 11-16 July 2010.

examples of a broader textual inference problem, and in general such knowledge is quite difficult to acquire and apply. However, in the present case we can take advantage of event extraction to learn these rules in a simpler fashion, which we present below.

Most current event extraction systems are based on phrase or sentence level extraction. Several recent studies use high-level information to aid local event extraction systems. For example, Finkel et al. (2005), Maslennikov and Chua (2007), Ji and Grishman (2008), and Patwardhan and Riloff (2007, 2009) tried to use discourse, document, or cross-document information to improve information extraction.

However, most of this research focuses on single event extraction, or focuses on high-level information within a single event type, and does not consider information acquired from other event types. We extend these approaches by introducing cross-event information to enhance the performance of multi-event-type extraction systems. Cross-event information is quite useful: first, some events co-occur frequently, while other events do not. For example, *Attack*, *Die*, and *Injure* events very frequently occur together, while *Attack* and *Marry* are less likely to co-occur. Also, typical relations among the arguments of different types of events can be helpful in predicting information to be extracted. For example, the *Victim* of a *Die* event is probably the *Target* of the *Attack* event. As a result, we extend the observation that "a document containing a certain event is likely to contain more events of the same type", and base our approach on the idea that "a document containing a certain type of event is likely to contain instances of related events". In this paper, automatically extracted within-event and cross-event information is used to aid traditional sentence level event extraction.

2 Task Description

Automatic Content Extraction (ACE) defines an event as a specific occurrence involving participants[1], and it annotates 8 types and 33 subtypes of events. We first present some ACE terminology to understand this task more easily:

- **Entity**: an object or a set of objects in one of the semantic categories of interest, referred to in the document by one or more (coreferential) entity mentions.
- **Entity mention**: a reference to an entity (typically, a noun phrase)
- **Timex:** a time expression including date, time of the day, season, year, etc.
- **Event mention**: a phrase or sentence within which an event is described, including trigger and arguments. An event mention must have one and only one trigger, and can have an arbitrary number of arguments.
- **Event trigger**: the main word that most clearly expresses an event occurrence. An ACE event trigger is generally a verb or a noun.
- **Event mention arguments (roles)**[2]: the entity mentions that are involved in an event mention, and their relation to the event. For example, event *Attack* might include participants like *Attacker, Target*, or attributes like *Time_within* and *Place*. Arguments will be taggable only when they occur within the scope of the corresponding event, typically the same sentence.

Consider the sentence:

*(4) Three **murders** occurred in **France today**, including the senseless **slaying** of **Bob Cole** and the **assassination** of **Joe Westbrook**. **Bob** was on his way home when **he** was **attacked**…*

Event extraction depends on previous phases like name identification, entity mention classification and coreference. Table 1 shows the results of this preprocessing. Note that entity mentions that share the same EntityID are coreferential and treated as the same object.

Entity(Timex) mention	head word	Entity ID	Entity type
0001-1-1	France	0001-1	GPE
0001-T1-1	Today	0001-T1	Timex
0001-2-1	Bob Cole	0001-2	PER
0001-3-1	Joe Westbrook	0001-3	PER
0001-2-2	Bob	0001-2	PER
0001-2-3	He	0001-2	PER

Table 1. An example of entities and entity mentions and their types

[1] See http://projects.ldc.upenn.edu/ace/docs/English-Events-Guidelines_v5.4.3.pdf for a description of this task.

[2] Note that we do not deal with event mention coreference in this paper, so each event mention is treated as a separate event.

There are three *Die* events, which share the same *Place* and *Time* roles, with different *Victim* roles. And there is one *Attack* event sharing the same *Place* and *Time* roles with the *Die* events.

Event type	Trigger	Role		
		Place	**Victim**	**Time**
Die	murder	0001-1-1		0001-T1-1
Die	death	0001-1-1	0001-2-1	0001-T1-1
Die	killing	0001-1-1	0001-3-1	0001-T1-1
Event type	**Trigger**	**Role**		
		Place	**Target**	**Time**
Attack	attack	0001-1-1	0001-2-3	0001-T1-1

Table2. An example of event trigger and roles

In this paper, we treat the 33 event subtypes as separate event types and do not consider the hierarchical structure among them.

3 Related Work

Almost all the current ACE event extraction systems focus on processing one sentence at a time (Grishman et al., 2005; Ahn, 2006; Hardy et al. 2006). However, there have been several studies using high-level information from a wider scope:

Maslennikov and Chua (2007) use discourse trees and local syntactic dependencies in a pattern-based framework to incorporate wider context to refine the performance of relation extraction. They claimed that discourse information could filter noisy dependency paths as well as increasing the reliability of dependency path extraction.

Finkel et al. (2005) used Gibbs sampling, a simple Monte Carlo method used to perform approximate inference in factored probabilistic models. By using simulated annealing in place of Viterbi decoding in sequence models such as HMMs, CMMs, and CRFs, it is possible to incorporate non-local structure while preserving tractable inference. They used this technique to augment an information extraction system with long-distance dependency models, enforcing label consistency and extraction template consistency constraints.

Ji and Grishman (2008) were inspired from the hypothesis of "One Sense Per Discourse" (Yarowsky, 1995); they extended the scope from a single document to a cluster of topic-related documents and employed a rule-based approach to propagate consistent trigger classification and event arguments across sentences and documents. Combining global evidence from related documents with local decisions, they obtained an appreciable improvement in both event and event argument identification.

Patwardhan and Riloff (2009) proposed an event extraction model which consists of two components: a model for sentential event recognition, which offers a probabilistic assessment of whether a sentence is discussing a domain-relevant event; and a model for recognizing plausible role fillers, which identifies phrases as role fillers based upon the assumption that the surrounding context is discussing a relevant event. This unified probabilistic model allows the two components to jointly make decisions based upon both the local evidence surrounding each phrase and the "peripheral vision".

Gupta and Ji (2009) used cross-event information within ACE extraction, but only for recovering implicit time information for events.

4 Motivation

We analyzed the sentence-level baseline event extraction, and found that many events are missing or spuriously tagged because the local information is not sufficient to make a confident decision. In some local contexts, it is easy to identify an event; in others, it is hard to do so. Thus, if we first tag the easier cases, and use such knowledge to help tag the harder cases, we might get better overall performance. In addition, global information can make the event tagging more consistent at the document level.

Here are some examples. For trigger classification:

*The pro-reform director of Iran's biggest-selling daily newspaper and official organ of Tehran's municipality has **stepped** down following the **appointment** of a conservative …it was **founded** a decade ago … but a conservative city council was **elected** in the February 28 municipal polls … Mahmud Ahmadi-Nejad, reported to be a hardliner among conservatives, was **appointed** mayor on Saturday …**Founded** by former mayor Gholamhossein Karbaschi, Hamshahri…*

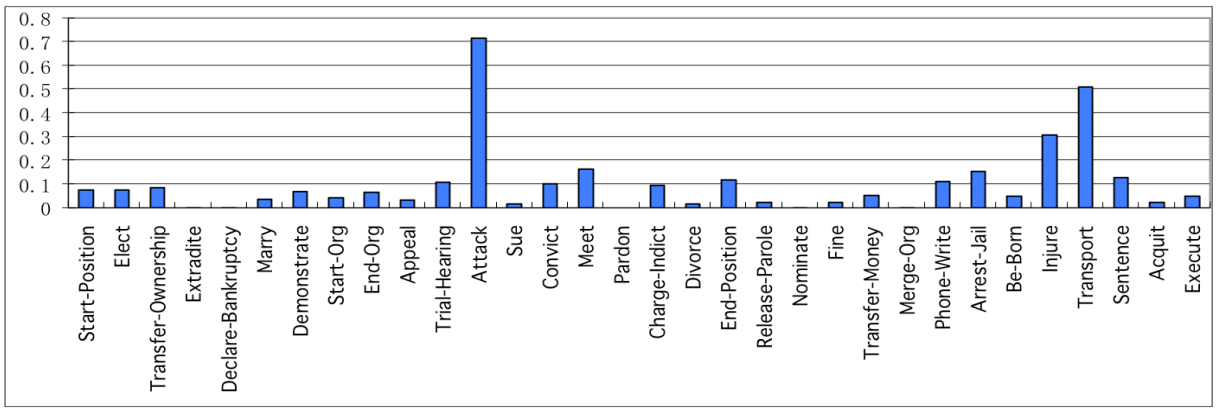

Figure 1. Conditional probability of the other 32 event types in documents where a *Die* event appears

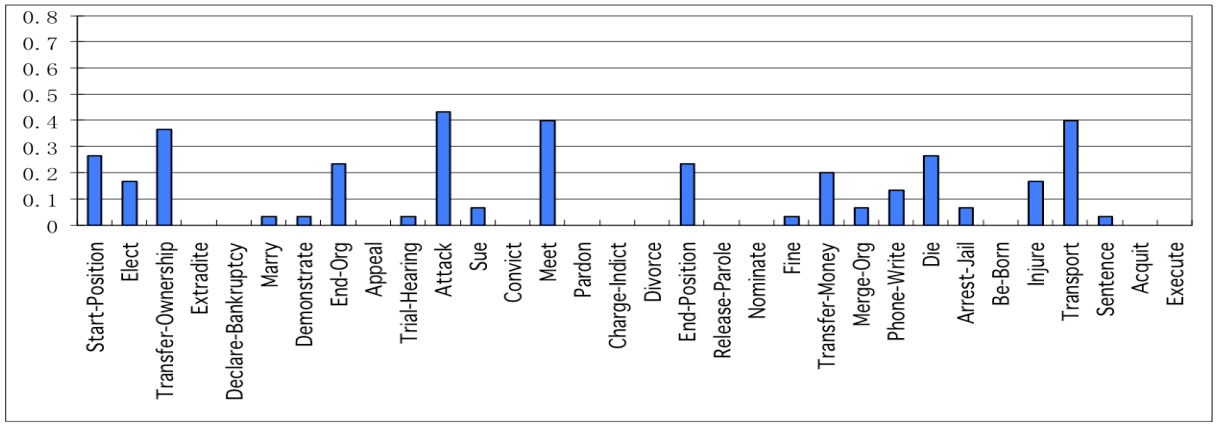

Figure 2. Conditional probability of the other 32 event types in documents where a *Start-Org* event appears

The sentence level baseline system finds event triggers like "*founded*" (trigger of *Start-Org*), "*elected*" (trigger of *Elect*), and "*appointment*" (trigger of *Start-Position*), which are easier to identify because these triggers have more specific meanings. However, it does not recognize the trigger "*stepped*" (trigger of *End-Position*) because in the training corpus "*stepped*" does not always appear as an *End-Position* event, and local context does not provide enough information for the MaxEnt model to tag it as a trigger. However, in the document that contains related events like *Start-Position*, "*stepped*" is more likely to be tagged as an *End-Position* event.

For argument classification, the cross-event evidence from the document level is also useful:

*British officials say they believe **Hassan** was a blindfolded woman seen being **shot** in the head by a hooded militant on a video obtained but not aired by the Arab television station Al-Jazeera. **She** would be the first foreign woman to **die** in the wave of kidnappings in Iraq…**she**'s been **killed** by (men in pajamas), turn Iraq upside down and find them.*

From this document, the local information is not enough for our system to tag "*Hassan*" as the target of an *Attack* event, because it is quite far from the trigger "*shot*" and the syntax is somewhat complex. However, it is easy to tag "*she*" as the *Victim* of a *Die* event, because it is the object of the trigger "*killed*". As "*she*" and "*Hassan*" are co-referred, we can use this easily tagged argument to help identify the harder one.

4.1 Trigger Consistency and Distribution

Within a document, there is a strong trigger consistency: if one instance of a word triggers an event, other instances of the same word will trigger events of the same type[3].

There are also strong correlations among event types in a document. To see this we calculated the conditional probability (in the ACE corpus) of a certain event type appearing in a document when another event type appears in the same document.

[3] This is true over 99.4% of the time in the ACE corpus.

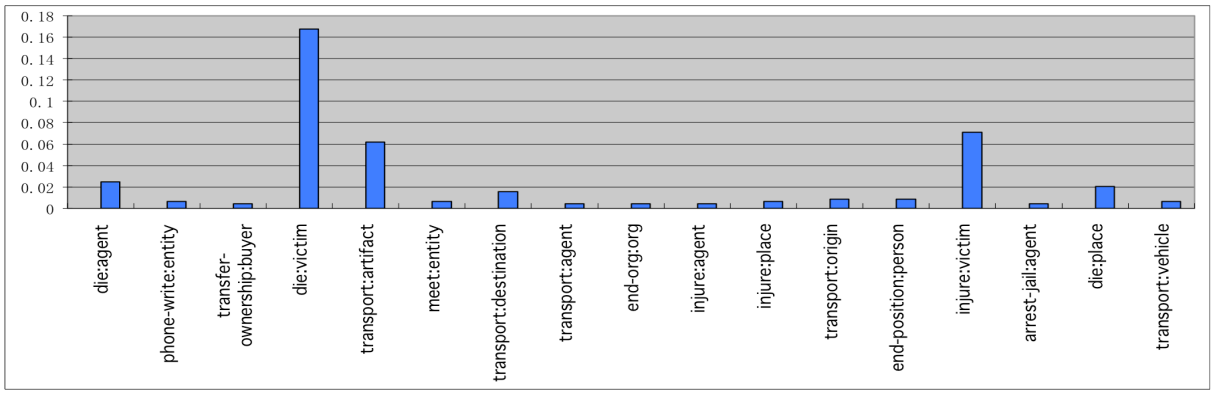

Figure 3. Conditional probability of all possible roles in other event types for entities that are the *Target*s of *Attack* events (roles with conditional probability below 0.002 are omitted)

Event	Cond. Prob.
Attack	0.714
Transport	0.507
Injure	0.306
Meet	0.164
Arrest-Jail	0.153
Sentence	0.126
Phone-Write	0.111
End-Position	0.116
Trial-Hearing	0.105
Convict	0.100

Table 3. Events co-occurring with *die* events with conditional probability > 10%

As there are 33 subtypes, there are potentially 33·32/2=528 event pairs. However, only a few of these appear with substantial frequency. For example, there are only 10 other event types that occur in more than 10% of the documents in which a *die* event appears. From Table 3, we can see that *Attack*, *Transport* and *Injure* events appear frequently with *Die*. We call these the related event types for *Die* (see Figure 1 and Table 3).

The same thing happens for *Start-Org* events, although its distribution is quite different from *Die* events. For *Start-Org*, there are more related events like *End-Org*, *Start-Position*, and *End-Position* (Figure 2). But there are 12 other event types which never appear in documents containing *Start-Org* events.

From the above, we can see that the distributions of different event types are quite different, and these distributions might be good predictors for event extraction.

4.2 Role Consistency and Distribution

Normally one entity, if it appears as an argument of multiple events *of the same type* in a single document, is assigned the same role each time.[4]

There is also a strong relationship between the roles when an entity participates in different types of events in a single document. For example, we checked all the entities in the ACE corpus that appear as the *Target* role for an *Attack* event, and recorded the roles they were assigned for other event types. Only 31 other event-role combinations appeared in total (out of 237 possible with ACE annotation), and 3 clearly dominated. In Figure 3, we can see that the most likely roles for the *Target* role of the *Attack* event are the *Victim* role of the *Die* or *Injure* event and the *Artifact* role of the *Transport* event. The last of these corresponds to troop movements prior to or in response to attacks.

5 Cross-event Approach

In this section we present our approach to using document-level event and role information to improve sentence-level ACE event extraction.

Our event extraction system is a two-pass system where the sentence-level system is first applied to make decisions based on local information. Then the confident local information is collected and gives an approximate view of the content of the document. The document level system is finally applied to deal with the cases which the local

[4] This is true over 97% of the time in the ACE corpus.

system can't handle, and achieve document consistency.

5.1 Sentence-level Baseline System

We use a state-of-the-art English IE system as our baseline (Grishman et al. 2005). This system extracts events independently for each sentence, because the definition of event mention argument constrains them to appear in the same sentence. The system combines pattern matching with statistical models. In the training process, for every event mention in the ACE training corpus, patterns are constructed based on the sequences of constituent heads separating the trigger and arguments. A set of Maximum Entropy based classifiers are also trained:

- **Argument Classifier:** to distinguish arguments of a potential trigger from non-arguments;
- **Role Classifier:** to classify arguments by argument role.
- **Reportable-Event Classifier (Trigger Classifier):** Given a potential trigger, an event type, and a set of arguments, to determine whether there is a reportable event mention.

In the test procedure, each document is scanned for instances of triggers from the training corpus. When an instance is found, the system tries to match the environment of the trigger against the set of patterns associated with that trigger. This pattern-matching process, if successful, will assign some of the mentions in the sentence as arguments of a potential event mention. The argument classifier is applied to the remaining mentions in the sentence; for any argument passing that classifier, the role classifier is used to assign a role to it. Finally, once all arguments have been assigned, the reportable-event classifier is applied to the potential event mention; if the result is successful, this event mention is reported.[5]

5.2 Document-level Confident Information Collector

To use document-level information, we need to collect information based on the sentence-level baseline system. As it is a statistically-based model, it can provide a value that indicates how likely it is that this word is a trigger, or that the mention is an argument and has a particular role. We want to see if this value can be trusted as a confidence score. To this end, we set different thresholds from 0.1 to 1.0 in the baseline system output, and only evaluate triggers, arguments or roles whose confidence score is above the threshold. Results show that as the threshold is raised, the precision generally increases and the recall falls. This indicates that the value is consistent and a useful indicator of event/argument confidence (see Figure 4).[6]

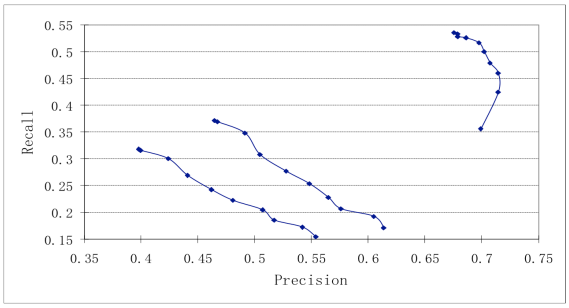

Figure 4. The performance of different confidence thresholds in the baseline system on the development set

To acquire confident document-level information, we only collect triggers and roles tagged with high confidence. Thus, a trigger threshold *t_threshold* and role threshold *r_threshold* are set to remove low confidence triggers and arguments. Finally, a table with *confident event* information is built. For every event, we collect its trigger and event type; for every argument, we use co-reference information and record every entity and its role(s) in events of a certain type.

To achieve document consistency, in cases where the baseline system assigns a word to triggers for more than one event type, if the margin between the probability of the highest and the second highest scores is above a threshold *m_threshold*, we only keep the event type with highest score and record this in the *confident-event* table. Otherwise (if the margin is smaller) the event type assignments will be recorded in a separate *conflict* table. The same strategy is applied to argument/role conflicts. We will not use information in the conflict table to infer the event type or argument/roles for other event mentions, because we cannot

[5] If the event arguments include some assigned by the pattern-matching process, the event mention is accepted unconditionally, bypassing the reportable- event classifier.

[6] The trigger classification curve doesn't follow the expected recall/precision trade-off, particularly at high thresholds. This is due, at least in part, to the fact that some events bypass the reportable-event classifier (trigger classifier) (see footnote 5). At high thresholds this is true of the bulk of the events.

confidently resolve the conflict. However, the event type and argument/role assignments in the conflict table will be included in the final output because the local confidence for the individual assignments is high.

As a result, we finally build two document-level confident-event tables: the event type table and the argument (role) table. A conflict table is also built but not used for further predictions (see Table 4).

Confident table		
Event type table		
Trigger	**Event Type**	
Met	Meet	
Exploded	Attack	
Went	Transport	
Injured	Injure	
Attacked	Attack	
Died	Die	
Argument role table		
Entity ID	**Event type**	**Role**
0004-T2	Die	Time Within
0004-6	Die	Place
0004-4	Die	Victim
0004-7	Die	Agent
0004-11	Attack	Target
0004-T3	Attack	Time Within
0004-12	Attack	Place
0004-10	Attack	Attacker
Conflict table		
Entity ID	**Event type**	**Roles**
0004-8	Attack	Victim, Agent

Table 4. Example of document-level confident-event table (event type and argument role entries) and conflict table

5.3 Statistical Cross-event Classifiers

To take advantage of cross-event relationships, we train two additional MaxEnt classifiers – a document-level trigger and argument classifier – and then use these classifiers to infer additional events and event arguments. In analyzing new text, the trigger classifier is first applied to tag an event, and then the argument (role) classifier is applied to tag possible arguments and roles of this event.

5.3.1 Document Level Trigger Classifier

From the document-level confident-event table, we have a rough view of what kinds of events are reported in this document. The trigger classifier predicts whether a word is the trigger of an event, and if so of what type, given the information (from the confident-event table) about other types of events in the document. Each feature of this classifier is the conjunction of:

- The base form of the word
- An event type
- A binary indicator of whether this event type is present elsewhere in the document

(There are 33 event types and so 33 features for each word).

5.3.2 Document Level Argument (Role) Classifier

The role classifier predicts whether a given mention is an argument of a given event and, if so, what role it takes on, again using information from the confident-event table about other events.

As noted above, we assume that the role of an entity is unique for a specific event type, although an entity can take on different roles for different event types. Thus, if there is a conflict in the document level table, the collector will only keep the one with highest confidence, or discard them all. As a result, every entity is assigned a unique role with respect to a particular event type, or *null* if it is not an argument of a certain event type.

Each feature is the conjunction of:

- The event type we are trying to assign an argument/role to.
- One of the 32 other event types
- The role of this entity with respect to the other event type elsewhere in the document, or *null* if this entity is not an argument of that type of event

5.4 Document Level Event Tagging

At this point, the low-confidence triggers and arguments (roles) have been removed and the document-level confident-event table has been built; the new classifiers are now used to *augment* the confident tags that were previously assigned based on local information.

For trigger tagging, we only apply the classifier to the words that do not have a confident local labeling; if the trigger is already in the document level confident-event table, we will not re-tag it.

performance / system/human	Trigger classification			Argument classification			Role classification		
	P	R	F	P	R	F	P	R	F
Sentence-level baseline system	67.56	53.54	59.74	46.45	37.15	41.29	41.02	32.81	36.46
Within-event-type rules	63.03	59.90	61.43	48.59	46.16	47.35	43.33	41.16	42.21
Cross-event statistical model	68.71	68.87	**68.79**	50.85	49.72	**50.28**	45.06	44.05	**44.55**
Human annotation1	59.2	59.4	59.3	60.0	69.4	64.4	51.6	59.5	55.3
Human annotation2	69.2	75.0	72.0	62.7	85.4	72.3	54.1	73.7	62.4

Table 5. Overall performance on blind test data

The argument/role tagger is then applied to all events—those in the confident-event table and those newly tagged. For argument tagging, we only consider the entity mentions in the same sentence as the trigger word, because by the ACE event guidelines, the arguments of an event should appear within the same sentence as the trigger. For a given event, we re-tag the entity mentions that have not already been assigned as arguments of that event by the confident-event or conflict table.

6 Experiments

We followed Ji and Grishman (2008)'s evaluation and randomly select 10 newswire texts from the ACE 2005 training corpora as our development set, which is used for parameter tuning, and then conduct a blind test on a separate set of 40 ACE 2005 newswire texts. We use the rest of the ACE training corpus (549 documents) as training data for both the sentence-level baseline event tagger and document-level event tagger.

To compare with previous work on within-event propagation, we reproduced Ji and Grishman (2008)'s approach for cross-sentence, within-event-type inference (see "within-event-type rules" in Table 5). We applied their within-document inference rules using the cross-sentence confident-event information. These rules basically serve to adjust trigger and argument classification to achieve document-wide consistency. This process treats each event type separately: information about events of a given type is used to infer information about other events of the same type.

We report the overall Precision (P), Recall (R), and F-Measure (F) on blind test data. In addition, we also report the performance of two human annotators on 28 ACE newswire texts (a subset of the blind test set).[7]

From the results presented in Table 5, we can see that using the document level cross-event information, we can improve the F score for trigger classification by 9.0%, argument classification by 9.0%, and role classification by 8.1%. Recall improved sharply, demonstrating that cross-event information could recover information that is difficult for the sentence-level baseline to extract; precision also improved over the baseline, although not as markedly.

Compared to the within-event-type rules, the cross-event model yields much more improvement for trigger classification: rule-based propagation gains 1.7% improvement while the cross-event model achieves a further 7.3% improvement. For argument and role classification, the cross-event model also gains 3% and 2.3% above that obtained by the rule-based propagation process.

7 Conclusion and Future Work

We propose a document-level statistical model for event trigger and argument (role) classification to achieve document level within-event and cross-event consistency. Experiments show that document-level information can improve the performance of a sentence-level baseline event extraction system.

The model presented here is a simple two-stage recognition process; nonetheless, it has proven sufficient to yield substantial improvements in event recognition and event

[7] The final key was produced by review and adjudication of the two annotations by a third annotator, which indicates that the event extraction task is quite difficult and human agreement is not very high.

argument recognition. Richer models, such as those based on joint inference, may produce even greater gains. In addition, extending the approach to cross-document information, following (Ji and Grishman 2008), may be able to further improve performance.

References

David Ahn. 2006. The stages of event extraction. In *Proc. COLING/ACL 2006 Workshop on Annotating and Reasoning about Time and Events.* Sydney, Australia.

J. Finkel, T. Grenager, and C. Manning. 2005. Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling. In *Proc. 43rd Annual Meeting of the Association for Computational Linguistics,* pages 363–370, Ann Arbor, MI, June.

Ralph Grishman, David Westbrook and Adam Meyers. 2005. NYU's English ACE 2005 System Description. In *Proc. ACE 2005 Evaluation Workshop*, Gaithersburg, MD.

Prashant Gupta, Heng Ji. 2009. Predicting Unknown Time Arguments based on Cross-Event Propagation. In *Proc. ACL-IJCNLP 2009.*

Hilda Hardy, Vika Kanchakouskaya and Tomek Strzalkowski. 2006. Automatic Event Classification Using Surface Text Features. In *Proc. AAAI06 Workshop on Event Extraction and Synthesis.* Boston, MA.

H. Ji and R. Grishman. 2008. Refining Event Extraction through Cross-Document Inference. In *Proc. ACL-08: HLT,* pages 254–262, Columbus, OH, June.

M. Maslennikov and T. Chua. 2007. A Multi resolution Framework for Information Extraction from Free Text. In *Proc. 45th Annual Meeting of the Association of Computational Linguistics,* pages 592–599, Prague, Czech Republic, June.

S. Patwardhan and E. Riloff. 2007. Effective Information Extraction with Semantic Affinity Patterns and Relevant Regions. In *Proc. Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, 2007,* pages 717–727, Prague, Czech Republic, June.

Patwardhan, S. and Riloff, E. 2009. A Unified Model of Phrasal and Sentential Evidence for Information Extraction. In *Proc. Conference on Empirical Methods in Natural Language Processing 2009, (EMNLP-09).*

David Yarowsky. 1995. Unsupervised Word Sense Disambiguation Rivaling Supervised Methods. In *Proc. ACL 1995.* Cambridge, MA.