Using Cross-Entity Inference to Improve Event Extraction

Yu Hong Jianfeng Zhang Bin Ma Jianmin Yao Guodong Zhou Qiaoming Zhu
School of Computer Science and Technology, Soochow University, Suzhou City, China
{hongy, jfzhang, bma, jyao, gdzhou, qmzhu}@suda.edu.cn

Abstract

Event extraction is the task of detecting certain specified types of events that are mentioned in the source language data. The state-of-the-art research on the task is transductive inference (e.g. cross-event inference). In this paper, we propose a new method of event extraction by well using cross-entity inference. In contrast to previous inference methods, we regard entity-type consistency as key feature to predict event mentions. We adopt this inference method to improve the traditional sentence-level event extraction system. Experiments show that we can get 8.6% gain in trigger (event) identification, and more than 11.8% gain for argument (role) classification in ACE event extraction.

1 Introduction

The event extraction task in ACE (Automatic Content Extraction) evaluation involves three challenging issues: distinguishing events of different types, finding the participants of an event and determining the roles of the participants.

The recent researches on the task show the availability of transductive inference, such as that of the following methods: cross-document, cross-sentence and cross-event inferences. Transductive inference is a process to use the known instances to predict the attributes of unknown instances. As an example, given a target event, the cross-event inference can predict its type by well using the related events co-occurred with it within the same document. 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. But cross-event inference can use a related event "*Then he went shopping*" within the same document to identify it as a *Transport* event correctly.

As the above example might suggest, the availability of transductive inference for event extraction relies heavily on the known evidences of an event occurrence in specific condition. However, the evidence supporting the inference is normally unclear or absent. For instance, the relation among events is the key clue for cross-event inference to predict a target event type, as shown in the inference process of the sentence (1). But event relation extraction itself is a hard task in Information Extraction. So cross-event inference often suffers from some false evidence (viz., misleading by unrelated events) or lack of valid evidence (viz., unsuccessfully extracting related events).

In this paper, we propose a new method of transductive inference, named cross-entity inference, for event extraction by well using the relations among entities. This method is firstly motivated by the inherent ability of entity types in revealing event types. From the sentences:

(2)He left the ***bathroom****.*
(3)He left ***Microsoft****.*

it is easy to identify the sentence (2) as a *Transport* event in ACE, which means that he left the place, because nobody would retire (*End-Position* type) from a bathroom. And compared to the entities in sentence (1) and (2), the entity "*Microsoft*" in (3) would give us more confidence to tag the "*left*" event as an *End-Position* type, because people are used to giving the full name of the place where they retired.

The cross-entity inference is also motivated by the phenomenon that the entities of the same type often attend similar events. That gives us a way to predict event type based on entity-type consistency. From the sentence:

*(4)****Obama*** *beats McCain.*

it is hard to identify it as an *Elect* event in ACE, which means Obama wins the Presidential Election,

Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics, pages 1127–1136,
Portland, Oregon, June 19-24, 2011.

or an *Attack* event, which means Obama roughs somebody up. But if we have the priori knowledge that the sentence "*Bush beats McCain*" is an *Elect* event, and "*Obama*" was a presidential contender just like "*Bush*" (strict type consistency), we have ample evidence to predict that the sentence (4) is also an *Elect* event.

Indeed above cross-entity inference for event-type identification is not the only use of entity-type consistency. As we shall describe below, we can make use of it at all issues of event extraction:

- For **event type**: the entities of the same type are most likely to attend similar events. And the events often use consistent or synonymous trigger.
- For **event argument (participant)**: the entities of the same type normally co-occur with similar participants in the events of the same type.
- For **argument role**: the arguments of the same type, for the most part, play the same roles in similar events.

With the help of above characteristics of entity, we can perform a step-by-step inference in this order:

- ***Step 1***: predicting event type and labeling trigger given the entities of the same type.
- ***Step 2***: identifying arguments in certain event given priori entity type, event type and trigger that obtained by step 1.
- ***Step 3***: determining argument roles in certain event given entity type, event type, trigger and arguments that obtained by step 1 and step 2.

On the basis, we give a blind cross-entity inference method for event extraction in this paper. In the method, we first regard entities as queries to retrieve their related documents from large-scale language resources, and use the global evidences of the documents to generate entity-type descriptions. Second we determine the type consistency of entities by measuring the similarity of the type descriptions. Finally, given the priori attributes of events in the training data, with the help of the entities of the same type, we perform the step-by-step cross-entity inference on the attributes of test events (candidate sentences).

In contrast to other transductive inference methods on event extraction, the cross-entity inference makes every effort to strengthen effects of entities in predicting event occurrences. Thus the inferential process can benefit from following aspects: 1) less false evidence, viz. less false entity-type consistency (the key clue of cross-entity inference), because the consistency can be more precisely determined with the help of fully entity-type description that obtained based on the related information from Web; 2) more valid evidence, viz. more entities of the same type (the key references for the inference), because any entity never lack its congeners.

2 Task Description

The event extraction task we addressing is that of the Automatic Content Extraction (ACE) evaluations, where an event is defined as a specific occurrence involving participants. And event extraction task requires that certain specified types of events that are mentioned in the source language data be detected. We first introduce 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 (co-referential) entity mentions.
- **Entity mention**: a reference to an entity (typically, a noun phrase).
- **Event trigger**: the main word that most clearly expresses an event occurrence (An ACE event trigger is generally a verb or a noun).
- **Event arguments**: the entity mentions that are involved in an event (viz., participants).
- **Argument roles**: the relation of arguments to the event where they participate.
- **Event mention**: a phrase or sentence within which an event is described, including trigger and arguments.

The 2005 ACE evaluation had 8 types of events, with 33 subtypes; for the purpose of this paper, we will treat these simply as 33 separate event types and do not consider the hierarchical structure among them. Besides, the ACE evaluation plan defines the following standards to determine the correctness of an event extraction:

- A trigger is correctly labeled if its event type and offset (viz., the position of the trigger word in text) match a reference trigger.
- An argument is correctly identified if its event type and offsets match any of the reference argument mentions, in other word, correctly recognizing participants in an event.
- An argument is correctly classified if its role matches any of the reference argument mentions.

Consider the sentence:

(5) It has refused in ***the last five years*** *to* ***revoke*** *the license of* ***a single doctor*** *for committing medical errors*.[1]

The event extractor should detect an *End-Position* event mention, along with the trigger word "*revoke*", the position "*doctor*", the person whose license should be revoked, and the time during which the event happened:

Event type	***End-Position***	
Trigger	***revoke***	
Arguments	***a single doctor***	Role=***Person***
	doctor	Role=***Position***
	the last five years	Role=***Time-within***

Table 1: Event extraction example

It is noteworthy that event extraction depends on previous phases like name identification, entity mention co-reference and classification. Thereinto, the name identification is another hard task in ACE evaluation and not the focus in this paper. So we skip the phase and instead directly use the entity labels provided by ACE.

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; Hardyet 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.

Liao and Grishman (2010) propose document level cross-event inference to improve event extraction. In contrast to Gupta's work, Liao do not limit themselves to time information for events, but rather use related events and event-type consistency to make predictions or resolve ambiguities regarding a given event.

4 Motivation

In event extraction, current transductive inference methods focus on the issue that many events are missing or spuriously tagged because the local information is not sufficient to make a confident decision. The solution is to mine credible evidences of event occurrences from global information and regard that as priori knowledge to predict unknown event attributes, such as that of cross-document and cross-event inference methods.

However, by analyzing the sentence-level baseline event extraction, we found that the entities within a sentence, as the most important local information, actually contain sufficient clues for event detection. It is only based on the premise that we know the backgrounds of the entities beforehand. For instance, if we knew the entity "*vesuvius*" is an active volcano, we could easily identify

[1] Selected from the file "CNN_CF_20030304.1900.02" in ACE-2005 corpus.

the word “*erupt*”, which co-occurred with the entity, as the trigger of a “*volcanic eruption*” event but not that of a “*spotty rash*”.

In spite of that, it is actually difficult to use an entity to directly infer an event occurrence because we normally don’t know the inevitable connection between the background of the entity and the event attributes. But we can well use the entities of the same background to perform the inference. In detail, if we first know *entity(a)* has the same background with *entity(b)*, and we also know that *entity(a)*, as a certain role, participates in a specific event, then we can predict that *entity(b)* might participtes in a similar event as the same role.

Consider the two sentences[2] from ACE corpus:

(5) ***American*** *case for* ***war*** *against* ***Saddam***.

(6) ***Bush*** *should* ***torture*** *the* ***al Qaeda chief operations officer***.

The sentences are two event mentions which have the same attributes:

(5)	Event type	***Attack***	
	Trigger	***war***	
	Arguments	***American***	Role=***Attacker***
		Saddam	Role=***Target***
(6)	Event type	***Attack***	
	Trigger	***torture***	
	Arguments	***Bush***	Role=***Attacker***
		...Qaeda chief ...	Role=***Target***

Table 2: Cross-entity inference example

From the sentences, we can find that the entities “*Saddam*” and “*Qaeda chief*” have the same background (viz., terrorist leader), and they are both the arguments of *Attack* events as the role of *Target*. So if we previously know any of the event mentions, we can infer another one with the help of the entities of the same background.

In a word, the cross-entity inference, we proposed for event extraction, bases on the hypothesis:

Entities of the consistent type normally participate in similar events as the same role.

As we will introduce below, some statistical data from ACE training corpus can support the hypothesis, which show the consistency of event type and role in event mentions where entities of the same type occur.

4.1 Entity Consistency and Distribution

Within the ACE corpus, there is a strong entity consistency: if one entity mention appears in a type of event, other entity mentions of the same type will appear in similar events, and even use the same word to trigger the events. To see this we calculated the conditional probability (in the ACE corpus) of a certain entity type appearing in the 33 ACE event subtypes.

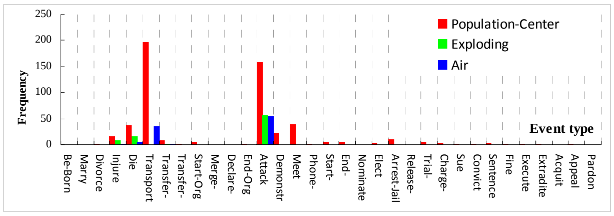

Figure 1. Conditional probability of a certain entity type appearing in the 33 ACE event subtypes (Here only the probabilities of *Population-Center*, *Exploding* and *Air* entities as examples)

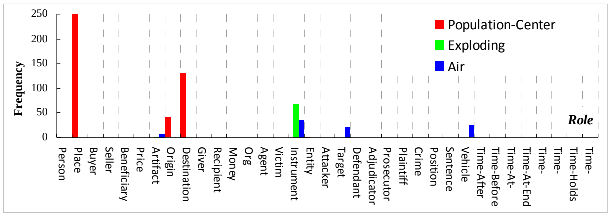

Figure 2. Conditional probability of an entity type appearing as the 34 ACE role types (Here only the probabilities of *Population-Center*, *Exploding* and *Air* entities as examples)

As there are 33 event subtypes and 43 entity types, there are potentially 33*43=1419 entity-event combinations. However, only a few of these appear with substantial frequency. For example, the *Population-Center* entities only occur in 4 types of event mentions with the conditional probability more than 0.05. From Table 3, we can find that only *Attack* and *Transport* events co-occur frequently with *Population-Center* entities (see Figure 1 and Table 3).

Event	**Cond.Prob.**	**Freq.**
Transport	0.368	197
Attack	0.295	158
Meet	0.073	39
Die	0.069	37

Table 3: Events co-occurring with *Population-Center* with the conditional probability > 0.05

Actually we find that most entity types appear in more restricted event mentions than *Population-Center* entity. For example, *Air* entity only co-occurs with 5 event types (*Attack*, *Transport*, *Die*, *Transfer-Ownership* and *Injure*), and *Exploding*

[2] They are extracted from the files “CNN_CF_20030305.1900.00-1” and “CNN_CF_20030303.1900.06-1” respectively.

entity co-occurs with 4 event types (see Figure 1). Especially, they only co-occur with one or two event types with the conditional probability more than 0.05.

	Evnt.<=5	5<Evnt.<=10	Evnt.>10
Freq. > 0	24	7	12
Freq. >10	37	4	2
Freq. >50	41	1	1

Table 4: Distribution of entity-event combination corresponding to different co-occurrence frequency

Table 4 gives the distributions of whole ACE entity types co-occurring with event types. We can find that there are 37 types of entities (out of 43 in total) appearing in less than 5 types of event mentions when entity-event co-occurrence frequency is larger than 10, and only 2 (e.g. *Individual*) appearing in more than 10 event types. And when the frequency is larger than 50, there are 41 (95%) entity types co-occurring with less than 5 event types. These distributions show the fact that most instances of a certain entity type normally participate in events of the same type. And the distributions might be good predictors for event type detection and trigger determination.

Air (Entity type)	
Attack event	***Fighter plane*** (subtype 1): "*MiGs*" "*enemy planes*" "*warplanes*" "*allied aircraft*" "*U.S. jets*" "*a-10 tank killer*" "*b-1 bomber*" "*a-10 warthog*" "*f-14 aircraft*" "*apache helicopter*"
Transport event	***Spacecraft*** (subtype 2): "*russian soyuz capsule*" "*soyuz*"
	Civil aviation (subtype 3): "*airliners*" "*the airport*" "*Hooters Air executive*"
	Private plane (subtype 4): "*Marine One*" "*commercial flight*" "*private plane*"

Table 5: Event types co-occurred with *Air* entities

Besides, an ACE entity type actually can be divided into more cohesive subtypes according to similarity of background of entity, and such a subtype nearly always co-occur with unique event type. For example, the *Air* entities can be roughly divided into 4 subtypes: *Fighter plane*, *Spacecraft*, *Civil aviation* and *Private plane*, within which the *Fighter plane* entities all appear in *Attack* event mentions, and other three subtypes all co-occur with *Transport* events (see Table 5). This consistency of entities in a subtype is helpful to improve the precision of the event type predictor.

4.2 Role Consistency and Distribution

The same thing happens for entity-role combinations: entities of the same type normally play the same role, especially in the event mentions of the same type. For example, the *Population-Center* entities occur in ACE corpus as only 4 role types: *Place*, *Destination*, *Origin* and *Entity* respectively with conditional probability 0.615, 0.289, 0.093, 0.002 (see Figure 2). And They mainly appear in *Transport* event mentions as *Place*, and in *Attack* as *Destination*. Particularly the *Exploding* entities only occur as *Instrument* and *Artifact* respectively with the probability 0.986 and 0.014. They almost entirely appear in *Attack* events as *Instrument*.

	Evnt.<=5	5<Evnt.<=10	Evnt.>10
Freq. > 0	32	5	6
Freq. >10	38	3	2
Freq. >50	42	1	0

Table 6: Distribution of entity-role combination corresponding to different co-occurrence frequency

Table 6 gives the distributions of whole entity-role combinations in ACE corpus. We can find that there are 38 entity types (out of 43 in total) occur as less than 5 role types when the entity-role co-occurrence frequency is larger than 10. There are 42 (98%) when the frequency is larger than 50, and only 2 (e.g. *Individual*) when larger than 10. The distributions show that the instances of an entity type normally occur as consistent role, which is helpful for cross-entity inference to predict roles.

5 Cross-entity Approach

In this section we present our approach to using blind cross-entity inference to improve sentence-level ACE event extraction.

Our event extraction system extracts events independently for each sentence, because the definition of event mention constrains them to appear in the same sentence. Every sentence that at least involves one entity mention will be regarded as a candidate event mention, and a randomly selected entity mention from the candidate will be the staring of the whole extraction process. For the entity mention, information retrieval is used to mine its background knowledge from Web, and its type is determined by comparing the knowledge with those in training corpus. Based on the entity type, the extraction system performs our step-by-step cross-entity inference to predict the attributes of the candidate event mention: trigger, event type, arguments, roles and whether or not being an event mention. The main frame of our event extraction system is shown in Figure 3, which includes both training and testing processes.

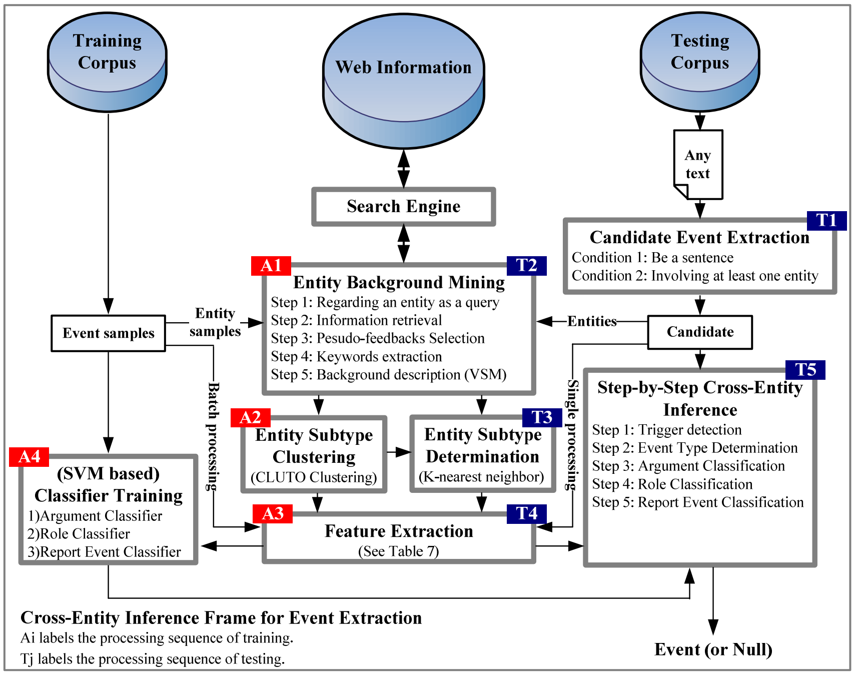
Training Corpus
Web Information
Testing Corpus
Any text
Search Engine
T1
Candidate Event Extraction
Condition 1: Be a sentence
Condition 2: Involving at least one entity
A1
T2
Entity Background Mining
Step 1: Regarding an entity as a query
Step 2: Information retrieval
Step 3: Pesudo-feedbacks Selection
Step 4: Keywords extraction
Step 5: Background description (VSM)
Event samples
Entity samples
Entities
Candidate
T5
Step-by-Step Cross-Entity Inference
Step 1: Trigger detection
Step 2: Event Type Determination
Step 3: Argument Classification
Step 4: Role Classification
Step 5: Report Event Classification
Batch processing
Single processing
A2
Entity Subtype Clustering
(CLUTO Clustering)
T3
Entity Subtype Determination
(K-nearest neighbor)
A4
(SVM based) Classifier Training
1)Argument Classifier
2)Role Classifier
3)Report Event Classifier
A3
T4
Feature Extraction
(See Table 7)
Cross-Entity Inference Frame for Event Extraction
Ai labels the processing sequence of training.
Tj labels the processing sequence of testing.
Event (or Null)

Figure 3. The frame of cross-entity inference for event extraction (including training and testing processes)

In the training process, for every entity type in the ACE training corpus, a clustering technique (CLUTO toolkit)[3] is used to divide it into different cohesive subtypes, each of which only contains the entities of the same background. For instance, the *Air* entities will be divided into *Fighter plane*, *Spacecraft*, *Civil aviation*, *Private plane*, etc (see Table 5). And for each subtype, we mine event mentions where this type of entities appear from ACE training corpus, and extract all the words which trigger the events to establish corresponding trigger list. Besides, a set of support vector machine (SVM) based classifiers are also trained:

- Argument Classifier: to distinguish arguments of a potential trigger from non-arguments[4];
- Role Classifier: to classify arguments by argument role;
- Reportable-Event Classifier (Trigger Classifier): Given entity types, a potential trigger, an event type, and a set of arguments, to determine whether there is a reportable event mention.

In the test process, for each candidate event mention, our event extraction system firstly predicts its triggers and event types: given an randomly selected entity mention from the candidate, the system determines the entity subtype it belonging to and the corresponding trigger list, and then all non-entity words in the candidate are scanned for a instance of triggers from the list. When an instance is found, the system tags the candidate as the event type that the most frequently co-occurs with the entity subtype in the events that triggered by the instance. Secondly the argument classifier is applied to the remaining mentions in the candidate; 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 candidate; if the result is successful, this event mention is reported.

5.1 Further Division of Entity Type

One of the most important pretreatments before our blind cross-entity inference is to divide the ACE entity type into more cohesive subtype. The greater consistency among backgrounds of entities in such a subtype might be good to improve the precision of cross-entity inference.

[3] http://oai.dtic.mil/oai/oai?verb=getRecord&metadataPrefix=html&identifier=ADA439508

[4] It is noteworthy that a sentence may include more than one event (more than one trigger). So it is necessary to distinguish arguments of a potential trigger from that of others.

For each ACE entity type, we collect all entity mentions of the type from training corpus, and regard each such mention as a query to retrieve the 50 most relevant documents from Web. Then we select 50 key words that the most weighted by TFIDF in the documents to roughly describe background of entity. After establishing the vector space model (VSM) for each entity mention of the type, we adopt a clustering toolkit (CLUTO) to further divide the mentions into different subtypes. Finally, for each subtype, we describe its centroid by using 100 key words which the most frequently occurred in relevant documents of entities of the subtype.

In the test process, for an entity mention in a candidate event mention, we determine its type by comparing its background against all centroids of subtypes in training corpus, and the subtype whose centroid has the most Cosine similarity with the background will be assigned to the entity. It is noteworthy that global information from the Web is only used to measure the entity-background consistency and not directly in the inference process. Thus our event extraction system actually still performs a sentence-level inference based on local information.

5.2 Cross-Entity Inference

Our event extraction system adopts a step-by-step cross-entity inference to predict event. As discussed above, the first step is to determine the trigger in a candidate event mention and tag its event type based on consistency of entity type. Given the domain of event mention that restrained by the known trigger, event type and entity subtype, the second step is to distinguish the most probable arguments that co-occurring in the domain from the non-arguments. Then for each of the arguments, the third step can use the co-occurring arguments in the domain as important contexts to predict its role. Finally, the inference process determines whether the candidate is a reportable event mention according to a confidence coefficient. In the following sections, we focus on introducing the three classifiers: argument classifier, role classifier and reportable-event classifier.

5.2.1 Cross-Entity Argument Classifier

For a candidate event mention, the first step gives its event type, which roughly restrains the domain of event mentions where the arguments of the candidate might co-occur. On the basis, given an entity mention in the candidate and its type (see the pretreatment process in section 5.1), the argument classifier could predict whether other entity mentions co-occur with it in such a domain, if yes, all the mentions will be the arguments of the candidate. In other words, if we know an entity of a certain type participates in some event, we will think of what entities also should participate in the event. For instance, when we know a *defendant* goes on trial, we can conclude that the *judge*, *lawyer* and *witness* should appear in *court*.

Argument Classifier
Feature 1: an event type (an event-mention domain)
Feature 2: an entity subtype
Feature 3: entity-subtype co-occurrence in domain
Feature 4: distance to trigger
Feature 5: distances to other arguments
Feature 6: co-occurrence with trigger in clause
Role Classifier
Feature 1 and ***Feature 2***
Feature 7: entity-subtypes of arguments
Reportable-Event Classifier
Feature 1
Feature 8: confidence coefficient of trigger in domain
Feature 9: confidence coefficient of role in domain

Table 7: Features selected for SVM-based cross-entity classifiers

A SVM-based argument classifier is used to determine arguments of candidate event mention. Each feature of this classifier is the conjunction of:

- The subtype of an entity
- The event type we are trying to assign an argument to
- A binary indicator of whether this entity subtype co-occurs with other subtypes in such an event type (There are 266 entity subtypes, and so 266 features for each instance)

Some minor features, such as another binary indicator of whether arguments co-occur with trigger in the same clause (see Table 7).

5.2.2 Cross-Entity Role Classifier

For a candidate event mention, the arguments that given by the second step (argument classifier) provide important contextual information for predicting what role the local entity (also one of the arguments) takes on. For instance, when *citizens* (Arg1) co-occur with *terrorist* (Arg2), most likely the role of Arg1 is *Victim*. On the basis, with the help of event type, the prediction might be more precise. For instance, if the Arg1 and Arg2 co-occur in an *Attack* event mention, we will have more confidence in the *Victim* role of Arg1.

Besides, as discussed in section 4, entities of the same type normally take on the same role in similar events, especially when they co-occur with similar arguments in the events (see Table 2). Therefore, all instances of co-occurrence model {entity subtype, event type, arguments} in training corpus could provide effective evidences for predicting the role of argument in the candidate event mention. Based on this, we trained a SVM-based role classifier which uses following features:

- Feature 1 and Feature 2 (see Table 7)
- Given the event domain that restrained by the entity and event types, an indicator of what subtypes of arguments appear in the domain. (266 entity subtypes make 266 features for each instance)

5.2.3 Reportable-Event Classifier

At this point, there are still two issues need to be resolved. First, some triggers are common words which often mislead the extraction of candidate event mention, such as "*it*", "*this*", "*what*", etc. These words only appear in a few event mentions as trigger, but when they once appear in trigger list, a large quantity of noisy sentences will be regarded as candidates because of their commonness in sentences. Second, some arguments might be tagged as more than one role in specific event mentions, but as ACE event guideline, one argument only takes on one role in a sentence. So we need to remove those with low confidence.

A confidence coefficient is used to distinguish the correct triggers and roles from wrong ones. The coefficient calculate the frequency of a trigger (or a role) appearing in specific domain of event mentions and that in whole training corpus, then combines them to represent its confidence degree, just like TFIDF algorithm. Thus, the more typical triggers (or roles) will be given high confidence. Based on the coefficient, we use a SVM-based classifier to determine the reportable events. Each feature of this classifier is the conjunction of:

- An event type (domain of event mentions)
- Confidence coefficients of triggers in domain
- Confidence coefficients of roles in the domain.

6 Experiments

We followed Liao (2010)'s evaluation and randomly select 10 newswire texts from the ACE 2005 training corpus 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 our event extraction system.

To compare with the reported work on cross-event inference (Liao, 2010) and its sentence-level baseline system, we cross-validate our method on 10 separate sets of 40 ACE texts, and report the optimum, worst and mean performances (see Table 8) on the data by using Precision (P), Recall (R) and F-measure (F). In addition, we also report the performance of two human annotators on 40 ACE newswire texts (a random blind test set): one knows the rules of event extraction; the other knows nothing about it.

6.1 Main Results

From the results presented in Table 8, we can see that using the cross-entity inference, we can improve the F score of sentence-level event extraction for trigger classification by 8.59%, argument classification by 11.86%, and role classification by 11.9% (mean performance). Compared to the cross-event inference, we gains 2.87% improvement for argument classification, and 3.81% for role classification (mean performance). Especially, our worst results also have better performances than cross-event inference.

Nonetheless, the cross-entity inference has worse F score for trigger determination. As we can see, the low Recall score weaken its F score (see Table 8). Actually, we select the sentence which at least includes one entity mention as candidate event mention, but lots of event mentions in ACE never include any entity mention. Thus we have missed some mentions at the starting of inference process.

In addition, the annotator who knows the rules of event extraction has a similar performance trend with systems: high for trigger classification, middle for argument classification, and low for role classification (see Table 8). But the annotator who never works in this field obtains a different trend: higher performance for argument classification. This phenomenon might prove that the step-by-step inference is not the only way to predicate event mention because human can determine arguments without considering triggers and event types.

Performance System/Human	Trigger (%)			Argument (%)			Role (%)		
	P	R	F	P	R	F	P	R	F
Sentence-level baseline	67.56	53.54	59.74	46.45	37.15	41.29	41.02	32.81	36.46
Cross-event inference	**68.71**	**68.87**	**68.79**	**50.85**	**49.72**	**50.28**	**45.06**	**44.05**	**44.55**
Cross-entity inference (optimum)	73.4	66.2	69.61	56.96	55.1	56	49.3	46.59	47.9
Cross-entity inference (worst)	71.3	64.17	66.1	51.28	50.3	50.78	46.3	44.3	45.28
Cross-entity inference (mean)	**72.9**	**64.3**	**68.33**	**53.4**	**52.9**	**53.15**	**51.6**	**45.5**	**48.36**
Human annotation 1 (blind)	58.9	59.1	59.0	62.6	65.9	64.2	50.3	57.69	53.74
Human annotation 2 (know rules)	74.3	76.2	75.24	68.5	75.8	71.97	61.3	68.8	64.86

Table 8: Overall performance on blind test data

6.2 Influence of Clustering on Inference

A main part of our blind inference system is the entity-type consistency detection, which relies heavily on the correctness of entity clustering and similarity measurement. In training, we used CLUTO clustering toolkit to automatically generate different types of entities based on their background-similarities. In testing, we use K-nearest neighbor algorithm to determine entity type.

Fighter plane (subtype 1 in ***Air*** entities): "*warplanes*" "*allied aircraft*" "*U.S. jets*" "*a-10 tank killer*" "*b-1 bomber*" "*a-10 warthog*" "*f-14 aircraft*" "*apache helicopter*" ***"terrorist" "Saddam" "Saddam Hussein" "Baghdad"…***

Table 9: Noises in subtype 1 of "*Air*" entities (The blod fonts are noises)

We obtained 129 entity subtypes from training set. By randomly inspecting 10 subtypes, we found nearly every subtype involves no less than 19.2% noises. For example, the subtype 1 of "*Air*" in Table 5 lost the entities of "*MiGs*" and "*enemy planes*", but involved "*terrorist*", "*Saddam*", etc (See Table 9). Therefore, we manually clustered the subtypes and retry the step-by-step cross-entity inference. The results (denoted as "Visible 1") are shown in Table 10, within which, we additionally show the performance of the inference on the rough entity types provided by ACE (denoted as "Visible 2"), such as the type of "*Air*", "*Population-Center*", "*Exploding*", etc., which normally can be divided into different more cohesive subtypes. And the "Blind" in Table 10 denotes the performances on our subtypes obtained by CLUTO.

It is surprised that the performances (see Table 10, F-score) on "Visible 1" entity subtypes are just a little better than "Blind" inference. So it seems that the noises in our blind entity types (CLUTO clusters) don't hurt the inference much. But by re-inspecting the "Visible 1" subtypes, we found that their granularities are not enough small: the 89 manual entity clusters actually can be divided into more cohesive subtypes. So the improvements of inference on noise-free "Visible 1" subtypes are partly offset by loss on weakly consistent entities in the subtypes. It can be proved by the poor performances on "Visible 2" subtypes which are much more general than "Visible 1". Therefore, a reasonable clustering method is important in our inference process.

F-score	**Trigger**	**Argument**	**Role**
Blind	68.33	53.15	48.36
Visible 1	69.15	53.65	48.83
Visible 2	51.34	43.40	39.95

Table 10: Performances on visible VS blind

7 Conclusions and Future Work

We propose a blind cross-entity inference method for event extraction, which well uses the consistency of entity mention to achieve sentence-level trigger and argument (role) classification. Experiments show that the method has better performance than cross-document and cross-event inferences in ACE event extraction.

The inference presented here only considers the helpfulness of entity types of arguments to role classification. But as a superior feature, contextual roles can provide more effective assistance to role determination of local argument. For instance, when an *Attack* argument appears in a sentence, a *Target* might be there. So if we firstly identify simple roles, such as the condition that an argument has only a single role, and then use the roles as priori knowledge to classify hard ones, may be able to further improve performance.

Acknowledgments

We thank Ruifang He. And we acknowledge the support of the National Natural Science Foundation of China under Grant Nos. 61003152, 60970057, 90920004.

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.

Jenny Rose Finkel, Trond Grenager and Christopher 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.

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

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

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.

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

Shasha Liao and Ralph Grishman. 2010. Using Document Level Cross-Event Inference to Improve Event Extraction. *In Proc. ACL-2010,* pages 789-797, Uppsala, Sweden, July.

Mstislav Maslennikov and Tat-Seng 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.

Siddharth Patwardhan and Ellen 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.

Siddharth Patwardhan and Ellen Riloff. 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.