

Towards High Performance Multilingual Event Extraction: Language Specific Issue and Feature Exploration

Zheng Chen

City University of New York
New York, NY, 11367

zchen1@gc.cuny.edu

Heng Ji

hengji@cs.qc.cuny.edu

Abstract

We present an Information Extraction (IE) system that combines traditional IE techniques and cross-document event ranking. Our final goal is to provide the user with ranked events upon a query. The first step is to set up a trainable event extraction engine to glean all possible events from multiple documents. The second step is to construct our data warehouse with refined events. The third step is to build an IR-like engine to produce ranked events upon a user's query. As the first step, we developed a multilingual event extraction engine using a modularized approach. In this paper, we focus on Chinese event extraction. We point out a language specific issue in Chinese trigger labeling, and then commit to discussing the contributions of lexical, syntactic and semantic features applied in trigger labeling and argument labeling tasks. As a result, we achieved high performance comparable to state-of-the-art English event extraction.

1 Introduction

Information Extraction (IE) systems can identify 'facts' (entities, relations and events) of a particular type within individual documents, and so can unleash the knowledge embedded in texts for many domains. Event extraction – 'classical' information extraction – remains the most challenging task, because it's situated at the end of an IE pipeline and thus suffers from propagation of errors from name tagging, coreference resolution, time expression identification, etc., also event triggers and arguments can be expressed in paraphrased forms while one form can represent different event types.

In this paper, we present an incubating IE system which is named BLENDER. Our system is composed of a trainable event extraction engine, an event warehouse and an IR engine featured by event ranking. By developing a trainable event extraction engine, we can adapt our system to diverse applications with minimum changes in algorithms. By building a large-scale event warehouse, we can construct a large event network linked by *who*, *when*, *where*, *what* and *how*. By setting up a traditional IR-like engine, we provide an entry for the end users to surf the enriched event network.

The first step to build our final system is to set up a trainable event extraction engine. In this paper we developed a modularized multilingual event extraction engine and focused on Chinese event extraction. We explored effective lexical, syntactic and semantic features that were applied in trigger labeling and argument labeling. As a result we achieved high performance comparable to state-of-the-art English event extraction.

The remainder of the paper is organized as follows. Section 2 gives some previous work. Section 3 presents our incubating BLENDER system with a focus on describing the modularized design of a multilingual event extraction engine. Section 4 points out a language specific issue in Chinese trigger labeling and discusses two strategies of trigger labeling: word-based and character-based. Section 5 presents argument labeling. Section 6 discusses the experimental results. Section 7 concludes the paper.

2 Previous Work

The tasks of event extraction were first explored in the series of Message Understanding Conferences (MUCs) started from 1987. The events in MUCs

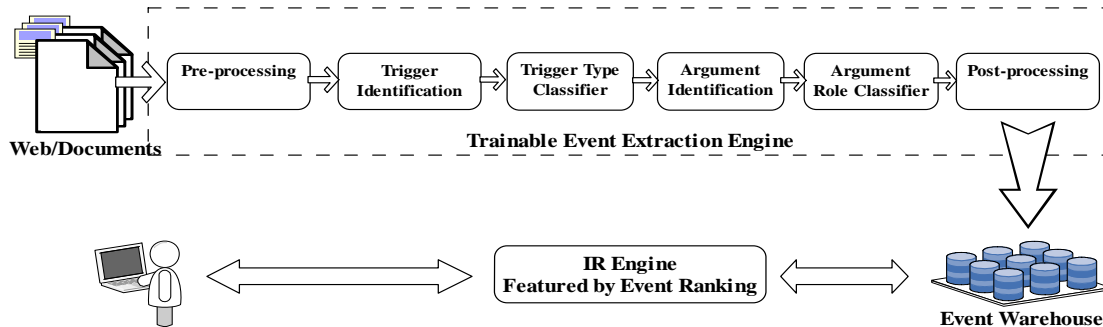


Figure 1. Framework of BLENDER System

were limited to finite topics, e.g., terrorist activities, management succession. The tasks were focused on filling the slots of an event such as the agent, the time and place, the effect, etc.

The Automatic Content Extraction¹ (ACE) program starting from 1999 extended the event extraction task to 8 event types (33 subtypes) from much wider sources such as newswire, broadcast conversation and weblogs. The ACE program defined the following terminology for event extraction task:

- Trigger: the word that most clearly expresses an event’s occurrence
- Argument: an entity mention, a time expression or value that plays a certain role in the event instance
- Event mention: a phrase or sentence with a distinguished trigger and arguments

For the following example,

Tom was born in 1990.

The event extraction system should identify “born” as trigger with the event type of “Be-Born”, meanwhile it should detect Tom and 1990 as arguments in which *Tom* plays a role as “Person” and *1990* plays a role as “Time-Within”.

Some English event extraction systems based on supervised learning have been reported by researchers (Ahn, 2006; Hardy et al., 2006; Ji and Grishman, 2008). However, to our knowledge, the language specific issue and feature contributions for non-English event extraction have not been reported by earlier researchers. It is worth noting that our event extraction task is based on ACE events.

3 BLENDER System Overview

The framework of BLENDER system is depicted in Figure 1. It takes raw documents as input and conducts some pre-processing steps. The texts are annotated with word segmentation, Part-of-Speech tags, parsing structures, entities, time expressions, and relations. The annotated documents are then sent to the following four components of the event extraction engine:

- Trigger Identification Classifier: while the system is scanning through sentences in the document, the classifier recognizes a word or a phrase as the trigger of a potential event mention.
- Trigger Type Classifier: it assigns an event type to the trigger. Once a trigger is found, the system starts constructing an event mention with arguments surrounding the trigger.
- Argument Identification Classifier: while the system is scanning through each entity mention/timex/value in the same sentence of the trigger, the classifier recognizes whether the entity mention/timex/value is an argument of the event mention.
- Argument Role Classifier: It assigns an argument role to the argument.

The framework of the event extraction engine itself is language independent. We developed the four classifiers based on Maximum Entropy method which is also language independent. However, when dealing with event extraction in a specific language, we will inevitably meet language specific issues and we should explore effective language-dependent and language-independent features to boost our final system.

The event warehouse and event IR engine are the other two important components in BLENDER system. However, they are not the emphasis of this paper. The event warehouse stores the event mentions and facilitates the queries of events by *who*,

¹ <http://www.nist.gov/speech/tests/ace/>

when, where, what and how. The IR engine applies an event-rank algorithm to produce ranked events.

4 Trigger Labeling

4.1 A Language-Specific Issue in Chinese Trigger Labeling

Chinese, and some other languages, e.g., Japanese do not have delimiters between words. Thus, segmentation is usually an indispensable step for further processing, e.g., Part-of-Speech tagging, parsing, etc. However, the segmentation may cause a problem in some tasks, e.g., name entity recognition (Jing et al., 2003) and event trigger identification. For a specific example, “*击毙*” (*shoot and kill*) is segmented as a Chinese word. However, there are two triggers in the word, one is “*击*” (*shoot*) with the event type of *Attack*, and the other is “*毙*” (*kill*) with the event type of *Die*. The trigger may also cross two or more words, e.g., the trigger is “*公开信*” (*public letter*) which crosses two words, “*公开*” (*public*) and “*信*” (*letter*).

In the ACE corpus, 2902 triggers exactly one-to-one match their corresponding words, meanwhile, 431 triggers are inconsistent with the words (either within the word, or across words). The inconsistency rate is as high as 13%. It is worth noting that segmentation does not affect argument labeling when the argument candidates are human-annotated entities, times, and values. However, the performance of trigger labeling can directly affect the final performance of argument labeling since arguments are found according to triggers.

4.2 Word-based Trigger Labeling

Trigger labeling includes trigger identification and trigger type classification. For simplicity, the two classifiers share the same sets of features.

4.2.1 Lexical Features

We apply the following lexical features for baseline word-based trigger labeling: word, POS tag of the word, previous word + word, word + next word, previous POS tag + POS tag, and POS tag + next POS tag.

4.2.2 Global Errata Table

We then construct an errata cache table to alleviate the word segmentation problem. In the training procedure, we use a hash table to record the inconsistencies between the trigger and the segmented

word. In the test procedure, if the scanned word has an entry in the errata table, we select the possible triggers in the entry as candidate triggers.

4.2.3 Semantic Dictionaries

We incorporate the following semantic dictionary resources: a full predicate list from Chinese Propbank (Xue and Palmer, 2008) and a Chinese synonym dictionary. We encode them as two features:

- **Predicate Existence:** this feature uses the combination of the trigger and a boolean value. If the trigger exists in the predicate list, the Boolean value is true, otherwise it is false, e.g. 成立_T
- **Synonym Entry:** this feature uses the combination of the trigger and its entry number in the synonym entry, e.g., 成立_Hc05A01. If the trigger does not have an entry, use “#” instead.

4.2.4 Syntactic Features

In addition we introduce the following syntax features from the parse tree:

- **Depth:** the depth of the trigger in the parse tree
- **Path to Root:** the path from the leaf node of the trigger to the root in the parse tree
- **Sub-categorization :** the phrase structure expanded by the father of the trigger
- **Phrase Type:** the phrase type of the trigger

4.2.5 Nearest Entity Information

In order to model the constraints of candidate arguments upon the candidate triggers, we also encode the following features based on entities in the context of the trigger:

- Entity type of the syntactically nearest entity mention in the parse tree
- Entity type of the physically nearest entity mention in the sentence

4.3 Character-based Trigger Labeling

Although the error table significantly helps to reduce segmentation inconsistencies, it is not a perfect solution since it only recognizes the inconsistencies existing in the training data.

To take a further step we build a separate character-based trigger identification classifier for comparison. We use a MEMM (Maximum Entropy Markov Model) to label each character with a tag indicating whether it is out of the trigger (O), or is the beginning of the trigger (B) or is a part of the trigger except the beginning (I). Our MEMM clas-

sifier performs sequential classification by assigning each character one of the three tags. We then apply Viterbi algorithm to decode the tag sequence and identify the triggers in the sequence.

Features used in our MEMM classifier include: the character, previous character, next character, previous tag and word-based features that the character carries. We apply the same sets of features for trigger type classification as used in word-based trigger labeling.

5 Argument Labeling

Argument labeling includes argument identification and role classification.

5.1 Baseline Features

We apply the following basic features: trigger, event subtype of the event mention, type of the ACE entity mention, head word of the entity mention, combined value of event subtype and head word, combined value of event subtype and entity subtype.

5.2 Neighbor words

Then we add the following lexical features based on the words close to a candidate argument:

- Left neighbor word of the entity mention, timex, or value
- Right neighbor word of the entity mention, timex, or value

5.3 Syntactic Features

In order to capture the interactions between candidate arguments and the identified trigger words, we designed the following syntactic features.

- **Sub-categorization:** the phrase structure expanding the parent of the trigger
- **Position:** the relative position of the entity mention/timex/value regarding to the trigger (before or after)
- **Path:** the minimal path from the entity mention/timex/value to the trigger.
- **Distance:** the shortest length from the entity mention/timex/value to the trigger in the parse tree

6 Experimental Results

6.1 Data and Scoring Metric

We used 2005 ACE training corpus for our experiment. The corpus contains 633 Chinese docu-

ments which are categorized by 3 genres: Newswire (238 documents), Broadcast News (298 documents) and Weblog (97 documents). In this paper we follow the setting of ACE diagnostic tasks and use the ground truth entities, times and values for our training and testing.

We randomly selected 558 documents as training set and 66 documents as test set. For the training set, we reserved 33 documents as development set.

We define the following standards to determine the *correctness* of an event mention:

- *A trigger is correctly labeled* if its event type and offsets match a reference trigger.
- *An argument is correctly identified* if its event type and offsets match any of the reference argument mentions.
- *An argument is correctly identified and classified* if its event type, offsets, and role match any of the reference argument mentions.

6.2 Overall System Performance

Table 1 shows the overall Precision (P), Recall (R) and F-Measure (F) scores of our baseline system, word-based system with integrated feature sets and character-based system with integrated feature sets. The fourth and fifth rows show the performance of two human annotators.

Comparing to a state-of-the-art English event tagger (we omitted references here for blind review), Chinese trigger labeling is 3% lower than English system while argument labeling is 3.5% higher.

6.3 Comparison between Word-based and Character-based Trigger Labeling

Table 1 also lists the comparison results between character-based and word-based trigger labeling. It indicates that character-based identification model outperforms the best word-based model (3.3% improvement in F-Measure) with precision as high as 82.4% (14.3% improvement), and a little loss in recall (2.1%). It is clear that MEMM classifier has strong ability in decoding correct triggers while it does not perform better in finding missing triggers when comparing with word-based ME classifier.

6.4 Feature Contributions for Word based Trigger Labeling

Performance System/Human	Trigger Identifica- tion			Trigger Identification +Classification			Argument Identification			Argument Identification +Classification		
	P	R	F	P	R	F	P	R	F	P	R	F
Baseline	61.0	50.0	54.9	58.7	48.2	52.9	49.5	38.2	43.1	44.6	34.4	38.9
Word-based Trigger Labeling	68.1	52.7	59.4	65.7	50.9	57.4	56.1	38.2	45.4	53.1	36.2	43.1
Character-based Trigger Labeling	82.4	50.6	62.7	78.8	48.3	59.9	64.4	36.4	46.5	60.6	34.3	43.8
Human Annotator1	80.2	79.6	79.9	75.2	74.6	74.9	70.4	73.1	71.7	58.6	60.9	59.7
Human Annotator2	83.5	81.0	82.2	82.7	80.3	81.5	79.3	82.6	80.9	66.8	69.6	68.2

Table 1. Overall system performance (%)

	Trigger Identification			Trigger Identification and Classification		
	P	R	F	P	R	F
Lexical features : (1)	61.0	50.0	54.9	58.7	48.2	52.9
(1) + Errata table: (2)	64.0	52.0	57.4	61.3	49.8	54.9
(2) + Dictionaries: (3)	64.9	53.5	58.6	62.7	51.6	56.6
(3)+ Syntactic features: (4)	64.3	51.8	57.4	60.6	48.9	54.1
(3) + Entity Information: (5)	68.1	52.7	59.4	65.7	50.9	57.4

Table 2. Feature Contributions for Word-based Trigger Labeling (%)

	Argument Identification			Argument Role Classification		
	P	R	F	P	R	F
Basic feature set: (1)	40.5	32.8	36.2	37.7	30.5	33.7
(1)+Left word: (2)	45.2	35.4	39.7	41.6	32.5	36.5
(1)+Right word: (3)	47.7	35.6	40.8	44.1	32.9	37.7
Feature set 2: (2)+(3)	49.0	35.7	41.3	46.1	33.6	38.9
(1)+Sub-categorization: (4)	41.9	33.1	37.0	38.7	30.5	34.1
(1)+Path: (5)	46.6	36.2	40.7	43.4	33.7	38.0
(1)+Distance: (6)	49.5	37.0	42.3	45.0	33.6	38.5
(1)+Position:(7)	43.8	35.3	39.1	41.0	33.1	36.6
Feature set 3 (from 4 to 7)	56.2	36.1	43.9	51.2	32.9	40.0
Total	56.1	38.2	45.4	53.1	36.2	43.1

Table 3. Feature Contributions for Argument Labeling (%)

Table 2 presents the feature contributions for word-based trigger labeling. It shows that the errata table improves all measures of trigger identification and trigger classification over the baseline. Thus, maintaining an errata table is an effective strategy. Table 2 also shows that semantic resources contribute to trigger labeling task.

It is worth noting that the performance drops when integrating the syntactic features. Our explanation might be that the trigger, unlike the predicate in the semantic role labeling task, can not only be a verb, but also can be a noun or even a compo-

nent in other types of phrases. Thus the syntactic position for the trigger in the parse tree is much more flexible than the predicate in Semantic Role Labeling. For this reason, syntactic features are not so discriminative to separating triggers from non-triggers. Furthermore, the syntactic features cannot discriminate the semantic meanings of a candidate trigger. In the following example,

S1: 运动员正在 **进入** 球场准备即将到来的球赛
The players are **entering** the stadium to prepare for the coming game.

S2: 很多农产品还没有 **进入** 市场就腐烂。

Many farm products have been rotted before entering the market.

The word “进入” (*entering*) in sentence 1 should be correctly recognized as a “Transport” event while it does not indicate an event in sentence 2. The phrase structures around the word “进入” in both sentences are exactly the same (VP→VP-NP). However, if an entity of “PERSON” appears ahead of “进入”, the word “进入” is much more likely to be a trigger.

Hence the nearby entity information could be effective. Table 2 shows that these two features enhance precision (3.2% improvement) with some loss in recall (0.8%) in trigger identification and enhance precision (3.0% improvement) with some loss in recall (0.7%) in type classification.

6.5 Feature Contributions for Argument Labeling

Table 3 shows that the two neighbor word features are fairly effective. We observe that in some patterns of event description, the left word is informative to tell the followed entity mention is an argument. For example, “被[Entity]打死”(killed by [Entity]) is a common pattern to describe an attack event, and the left neighbor word of the entity “被” (*by*) can strongly imply that the entity is an argument with a role of “Attacker”. Meanwhile, the right word can help to reduce the spurious arguments. For example, in the Chinese “的” (*of*) structure, the word “的” (*of*) strongly suggests that the entity on the left side of “的” is not an argument. For a specific example, in “纽约市的市长” (*The mayor of New York City*), “纽约市” (*New York City*) on the left side of “的” (*of*) cannot be considered as an argument.

The sub-categorization feature contributes little to the improvement since it is a feature shared by all the arguments in the parse tree. Table 3 also shows that Path and Distance are two effective features. It is obvious that in the parse tree, each argument attached to the trigger is in a certain syntactic configuration. For example, the path “NP↑VP↓VV” implies that it might be a Subject-Verb or Verb-Object structure and thus the entity in NP is highly likely to be an argument of the trigger (VV). Syntactic distance is a neat way to encode the relation between the argument and the

trigger without interference from symbolic expressions.

The path feature does not tell the relative position between the argument and the trigger. For example, in “Subject Verb Object” structure, the path from subject (entity) to verb (trigger) and the path from object (entity) to verb (trigger) might be exactly the same, but obviously the subject and the object have different argument roles. Hence, the position feature is helpful to discriminate argument roles in syntactically identical structure.

7 Conclusions and Future Work

In this paper, we took a close look at language specific issue in Chinese event extraction and explored effective features for Chinese event extraction task. All our work contributes to setting up a high performance multilingual event extraction engine.

For future work, we intend to explore an approach to conducting cross-lingual event extraction and investigate whether the cross-lingual inference can bootstrap either side when running two language event extraction engines in parallel. We will continue our work in setting up BLENDER system and carry out research in event ranking and event linking.

References

- David Ahn. 2006. The stages of event extraction. Proc. COLING/ACL 2006 Workshop on Annotating and Reasoning about Time and Events. Sydney, Australia.
- Hilda Hardy, Vika Kanchakouskaya and Tomek Strzalkowski. 2006. Automatic Event Classification Using Surface Text Features. Proc. AAAI06 Workshop on Event Extraction and Synthesis. Boston, Massachusetts. US.
- Heng Ji and Ralph Grishman. 2008. Refining Event Extraction Through Cross-document Inference. Proc. ACL 2008. Ohio, USA.
- Hongyan Jing, Radu Florian, Xiaoqiang Luo, Tong Zhang, and Abraham Ittycheriah. 2003. HowtogetaChineseName(Entity): Segmentation and combination issues. Proc. EMNLP 2003.
- Nianwen Xue and Martha Palmer. 2008. Adding Semantic Role to the Chinese Treebank. Natural Language Engineering. Cambridge University Press.