RBPB: Regularization-Based Pattern Balancing Method for Event Extraction

Lei Sha[1,2], Jing Liu[2], Chin-Yew Lin[2], Sujian Li[1], Baobao Chang[1], Zhifang Sui[1]
[1] Key Laboratory of Computational Linguistics, Ministry of Education
School of Electronics Engineering and Computer Science, Peking University
[2]Knowledge Computing Group, Microsoft Research
[1]{shalei, lisujian, chbb, szf}@pku.edu.cn
[2]{liudani,cyl}@microsoft.com

Abstract

Event extraction is a particularly challenging information extraction task, which intends to identify and classify event triggers and arguments from raw text. In recent works, when determining event types (trigger classification), most of the works are either pattern-only or feature-only. However, although patterns cannot cover all representations of an event, it is still a very important feature. In addition, when identifying and classifying arguments, previous works consider each candidate argument separately while ignoring the relationship between arguments. This paper proposes a **R**egularization-**B**ased **P**attern **B**alancing Method (RBPB). Inspired by the progress in representation learning, we use trigger embedding, sentence-level embedding and pattern features together as our features for trigger classification so that the effect of patterns and other useful features can be balanced. In addition, RBPB uses a regularization method to take advantage of the relationship between arguments. Experiments show that we achieve results better than current state-of-art equivalents.

1 Introduction

Event extraction has become a popular research topic in the area of information extraction. ACE 2005 defines event extraction task[1] as three sub-tasks: identifying the trigger of an event, identifying the arguments of the event, and distinguishing their corresponding roles. As an example in Figure 1, there is an "Attack" event triggered by "tear through" with three arguments. Each argument has one role.

In the trigger classification stage, some previous approaches (Grishman et al., 2005; Ji and Grishman, 2008; Liao and Grishman, 2010; Huang and Riloff, 2012) use patterns to decide the types of event triggers. However, pattern-based approaches suffer from low recall since real world events usually have a large variety of representations. Some other approaches (Hong et al., 2011; Li et al., 2013; Lu and Roth, 2012) identify and classify event triggers using a large set of features without using patterns. Although these features can be very helpful, patterns are still indispensable in many cases because they can identify a trigger with the correct event type with more than 96% accuracy according to our data analysis on ACE 2005 data sets.

In argument identification and classification, most approaches identify each candidate argument separately without considering the relation between arguments. We define two kinds of argument relations here: (1) Positive correlation: if one candidate argument belongs to one event, then the other is more likely to belong to the same event. For example, in Figure 1, the entity "a waiting shed" shares a common dependency head "tore" with "a powerful bomb", so when the latter entity is identified as an argument, the former is more likely to be identified. (2) Negative correlation: if one candidate argument belongs to one event, then the other is less likely to belong to the same event. For example, in Figure 1, "bus" is irrelevant to other arguments, so if other entities are identified as arguments "bus" is less likely to be identified. Note that although all the above relation examples have something to do with dependency analysis, the positive/negative relationship depends not only on dependency parsing, but many other aspects as well.

[1] http://www.itl.nist.gov/iad/mig/tests/ace/2005/

Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, pages 1224–1234,
Berlin, Germany, August 7-12, 2016. ©2016 Association for Computational Linguistics

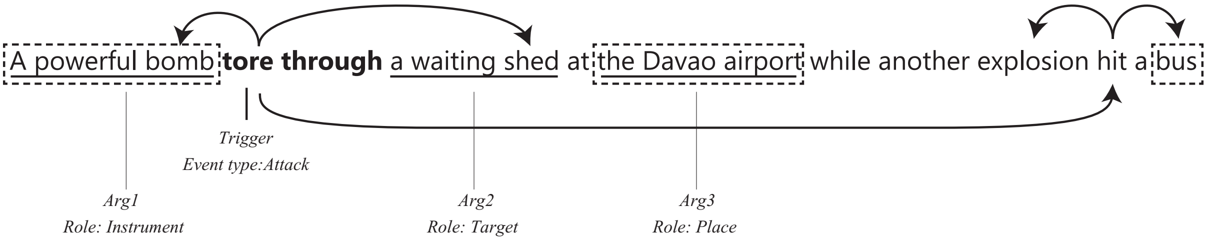


Figure 1: Event example: This is an event trigger by “tear through” with three arguments

In this paper, we propose using both patterns and elaborately designed features simultaneously to identify and classify event triggers. In addition, we propose using a regularization method to model the relationship between candidate arguments to improve the performance of argument identification. Our method is called Regularization-Based Pattern Balancing Method method.

The contributions of this paper are as follows:

- Inspired by the progress of representation learning, we use trigger embedding, sentence-level embedding, and pattern features together as the our features for balancing.
- We proposed a regularization-based method in order to make use of the relationship between candidate arguments. Our experiments on the ACE 2005 data set show that the regularization method does improve the performance of argument identification.

2 Related Work

There is a large body of previous work devoted to event extraction. Many traditional works focus on using pattern based methods for identifying event type (Kim and Moldovan, 1993; Riloff and others, 1993; Soderland et al., 1995; Huffman, 1996; Freitag, 1998b; Ciravegna and others, 2001; Califf and Mooney, 2003; Riloff, 1996; Riloff et al., 1999; Yangarber et al., 2000; Sudo et al., 2003; Stevenson and Greenwood, 2005; Grishman et al., 2005; Ji and Grishman, 2008; Liao and Grishman, 2010; Huang and Riloff, 2012). (Shinyama and Sekine, 2006; Sekine, 2006) are unsupervised methods of extracting patterns from open domain texts. Pattern is not always enough, although some methods (Huang and Riloff, 2012; Liu and Strzalkowski, 2012) use bootstrapping to get more patterns.

There are also feature-based classification methods (Freitag, 1998a; Chieu and Ng, 2002; Finn and Kushmerick, 2004; Li et al., 2005; Yu et al., 2005). Apart from the above methods, weakly supervised training (pattern-based and rule-based) of event extraction systems have also been explored (Riloff, 1996; Riloff et al., 1999; Yangarber et al., 2000; Sudo et al., 2003; Stevenson and Greenwood, 2005; Patwardhan and Riloff, 2007; Chambers and Jurafsky, 2011). In some of these systems, human work is needed to delete some nonsense patterns or rules. Other methods (Gu and Cercone, 2006; Patwardhan and Riloff, 2009) consider broader context when deciding on role fillers. Other systems take the whole discourse feature into consideration, such as (Maslennikov and Chua, 2007; Liao and Grishman, 2010; Hong et al., 2011; Huang and Riloff, 2011). Ji and Grishman (2008) even consider topic-related documents, proposing a cross-document method. (Liao and Grishman, 2010; Hong et al., 2011) use a series of global features (for example, the occurrence of one event type lead to the occurrence of another) to improve role assignment and event classification performance. Joint models (Li et al., 2013; Lu and Roth, 2012) are also considered an effective solution. (Li et al., 2013) make full use of the lexical and contextual features to get better results. The semi-CRF based method (Lu and Roth, 2012) trains separate models for each event type, which requires a lot of training data.

The dynamic multi-pooling convolutional neural network (DMCNN) (Chen et al., 2015) is currently the only widely used deep neural network based approach. DMCNN is mainly used to model contextual features. However, DMCNN still does not consider argument-argument interactions.

In summary, most of the above works are either pattern-only or features-only. Moreover, all of these methods consider arguments separately while ignoring the relationship between arguments, which is also important for argument identification. Even the joint method (Li et al., 2013) does not model argument relations directly. We use trigger embedding, sentence-level embedding, and pattern features together as our features for trigger classification and design a regularization-based method to solve the two problems.

3 ACE Event Extraction Task

Automatic Content Extraction (ACE) is an event extraction task. It annotates 8 types and 33 subtypes of events. ACE defines the following terminologies:

- Entity: an object or a set of objects in one of the semantic categories of interest
- Entity mention: a reference to an entity, usually a noun phrase (NP)
- Event trigger: the main word which most clearly expresses an event occurrence
- Event arguments: the entity mentions that are involved in an event
- Argument roles: the relation of arguments to the event where they participate, with 35 total possible roles
- Event mention: a phrase or sentence within which an event is described, including trigger and arguments

Given an English document, an event extraction system should identify event triggers with their subtypes and arguments from each sentence. An example is shown in Figure 1. There is an "Attack" event triggered by "tear through" with three arguments. Each argument has a role type such as "Instrument", "Target", etc.

For evaluation, we follow previous works (Ji and Grishman, 2008; Liao and Grishman, 2010; Li et al., 2013) to use the following criteria to determine the correctness of the predicted event mentions.

- A trigger is considered to be correct if and only if its event type and offsets (position in the sentence) can match the reference trigger;
- An argument is correctly identified if and only if its event type and offsets can match any reference arguments;
- An argument is correctly identified and classified if and only if its event type, offsets, and role match any of the reference arguments.

4 Baseline: JET Extractor for Events

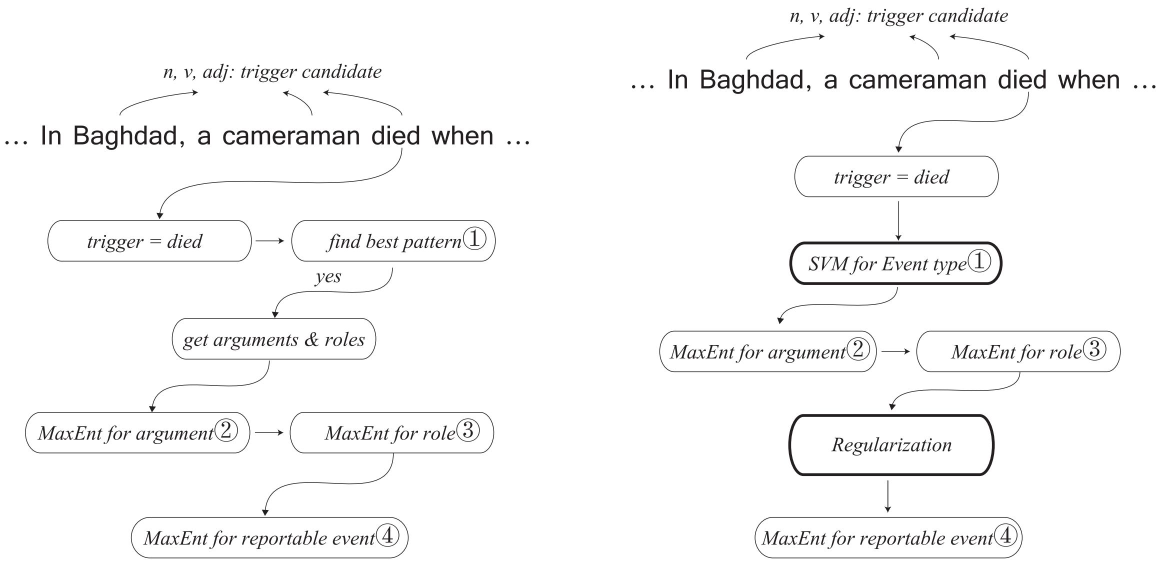
Many previous works take JET as their baseline system, including (Ji and Grishman, 2008), (Liao and Grishman, 2010), (Li et al., 2013). JET extracts events independently for each sentence. This system uses pattern matching to predict trigger and event types, then uses statistical modeling to identify and classify arguments. For each event mention in the training corpus of ACE, the patterns are constructed based on the sequences of constituent heads separating the trigger and arguments. After that, three Maximum Entropy classifiers are trained using lexical features.

- Argument Classifier: to distinguish arguments from non-arguments
- Role Classifier: to label arguments with an argument role
- Reportable-Event Classifier: to determine whether there is a reportable event mentioned (worth being taken as an event mention) according to the trigger, event type, and a set of arguments

Figure 2(a) shows the whole test procedure. In the test procedure, each sentence is scanned for nouns, verbs and adjectives as trigger candidates. When a trigger candidate is found, the system tries to match the context of the trigger against the set of patterns associated with that trigger. If this pattern matching process is successful, the best pattern will assign some of the entity mentions in the sentence as arguments of a potential event mention. Then JET uses the argument classifier to judge if the remaining entity mentions should also be identified. If yes, JET uses the role classifier to assign it a role. Finally, the reportable-event classifier is applied to decide whether this event mention should be reported.

5 Regularization-Based Pattern Balancing Method

Different with JET, as illustrated in Figure 2(b), our work introduces two major improvements: (1) balance the effect of patterns and other features (2)


n, v, adj: trigger candidate
... In Baghdad, a cameraman died when ...
trigger = died
find best pattern①
yes
get arguments & roles
MaxEnt for argument②
MaxEnt for role③
MaxEnt for reportable event④
n, v, adj: trigger candidate
... In Baghdad, a cameraman died when ...
trigger = died
SVM for Event type①
MaxEnt for argument②
MaxEnt for role③
Regularization
MaxEnt for reportable event④

(a) The flow chart of JET

(b) The flow chart of our approach

Figure 2: The left is the flow chart for JET. The right is the flow chart for our approach. The thick line block is our contribution

use a regularization-based method to make full use of the relation between candidate arguments.

The thick-edge blocks in Figure 2(b) represent our improvements. Since JET only uses patterns when predicting the event type, we use a SVM classifier to decide each candidate trigger's event type (classify the trigger). This classifier uses trigger embedding, sentence-level embedding and pattern features together for balancing. After the outputs of argument and role classifier are calculated, we make use of the argument relationship to regularize for a better result.

5.1 Balancing the Pattern effects

Deciding the event type is the same as classifying an event trigger. JET only uses patterns in this step: for a candidate trigger, we find that the best matched pattern and the corresponding event type are assigned to this trigger. We propose using feature-based methods while not ignoring the effect of patterns. Inspired by progress in representation learning, we use trigger embedding, sentence-level embedding and pattern embedding together as our features.

A pattern example is as follows:

(weapon) tore [through] (building) at (place) ⇒ Attack{Roles...}

where each pair of round brackets represents an entity and the word inside is one of the 18 entity types defined by UIUC NER Tool[2]. The word in the square brackets can choose to exist or not. After the right arrow there is an event schema, which can tell us what kind of event this is and which roles each entity should take.

Each pattern has a corresponding event type. A candidate trigger may match more than one pattern so that it has an event type distribution. Assume that there are N_T event types in total, we denote the pattern feature vector (namely, the event type's probability distribution calculated by the trigger's pattern set) as $P_E \in \mathbb{R}^{N_T}$, which is calculated by Eq 1.

$$P_E(i) = \frac{\#(\text{matched patterns of event type } i)}{\#(\text{all matched patterns})} \qquad (1)$$

Trigger embeddings are obtained using WORD2VEC[3] with the default "text8" training text data with length 200.

Since all of the NPs are potential roles in the event, they must contain the main information of the event. We extract all the NPs in the sentence and take the average word embedding of these NPs' head word as the sentence-level embedding. For example, in Figure 1, these NPs' head words are bomb, shed, and airport.

Pattern feature vectors, as distributions of event types over patterns, are also composed using

[2] http://cogcomp.cs.illinois.edu/page/software_view/NETagger

[3] http://code.google.com/p/word2vec/

continuous real values, which allows them to be viewed as a kind of pattern embedding and treated similarly to trigger and sentence embedding.

5.2 Capturing the Relationship Between Arguments

We find that there are two typical relations between candidate arguments: (1) positive correlation: if one candidate argument belongs to one event, then the other is more likely to belong to the same event; (2) negative correlation: if one candidate argument belongs to one event, then the other is less likely to belong to the same event.

We calculate a score for all the candidate arguments in a sentence to judge the quality of the argument identification and classification. For capturing the two kinds of relations, we intend to make that (1) the more positive relations the chosen arguments have, the higher the score is; (2) the more negative relations the chosen arguments have, the lower the score is.

For a trigger, if there are n candidate arguments, we set a $n \times n$ matrix C to represent the relationship between arguments. If $C_{i,j} = 1$, then argument i and argument j should belong to the same event. If $C_{i,j} = -1$, then argument i and argument j cannot belong to the same event. We will illustrate how to get matrix C in the next section.

We use a n-dim vector X to represent the identification result of arguments. Each entry of X is 0 or 1. 0 represents “noArg”, 1 represents “arg”. X can be assigned by maximizing $E(X)$ as defined by Eq 2.

$$
\begin{aligned}
X &= \operatorname*{argmax}_{X} E(X) \\
E(X) &= \lambda_1 X^T C X + \lambda_2 P^{arg}_{sum} \\
&\quad + (1 - \lambda_1 - \lambda_2) P^{role}_{sum}
\end{aligned} \tag{2}
$$

Here, $X^T C X$ means adding up all the relationship values if the two arguments are identified. Hence, the more the identified arguments are related, the larger the value $X^T C X$ is. P^{arg}_{sum} is the sum of all chosen arguments’ probabilities. The probability here is the output of the arguments’ maximum entropy classifier. P^{role}_{sum} is the sum of all the classified roles’ probabilities. The probability here is the output of the roles’ maximum entropy classifier.

Eq 2 shows that while we should identify and classify the candidate arguments with a larger probability, the argument relationship evaluation should also be as large as possible. The arguments should also follow the following constraints. These constraints together with Eq 2 can make the argument identification and classification help each other for a better result.

- Each entity can only take one role
- Each role can belong to one or more entities
- The role assignment must follow the event schema of the corresponding type, which means that only the roles in the event schema can occur in the event mention

We use the Beam Search method to search for the optimal assignment X as is shown in Algorithm 1. The hyperparameters λ_1 and λ_2 can be chosen according to development set.

```
Input: Argument relationship matrix: $C$
    the argument probabilities required by $P^{arg}_{sum}$
    the role probabilities required by $P^{role}_{sum}$
Data: K: Beam size
    n: Number of candidate arguments
Output: The best assignment $X$
Set beam $\mathcal{B} \leftarrow [\epsilon]$ ;
for $i \leftarrow 1 \cdots n$ do
    buf$\leftarrow \{z' \circ l | z' \in \mathcal{B}, l \in \{0, 1\}\}$;
    $\mathcal{B} \leftarrow [\epsilon]$ ;
    while $j \leftarrow 1 \cdots K$ do
        $x_{best} = \text{argmax}_{x \in \text{buf}} E(x)$;
        $\mathcal{B} \leftarrow \mathcal{B} \cup \{x_{best}\}$;
        buf$\leftarrow$buf$-\{x_{best}\}$;
    end
end
Sort $\mathcal{B}$ descendingly according to $E(X)$;
return $\mathcal{B}$[0];
```

Algorithm 1: Beam Search decoding algorithm for event extraction. $\circ$ means to concatenate an element to the end of a vector.

5.2.1 Training the Argument Relationship Structure

The argument relationship matrix C is very important in the regularization process. We train a maximum entropy classifier to predict the connection between two entities. We intend to classify the entity pairs into three classes: positive correlation, negative correlation, and unclear correlation. The entity pairs in the ground truth events (in training data) are used for our training data. We choose the following features:

- TRIGGER: the trigger of the event. The whole model is a pipelined model, so when classifying the argument relationship, the trigger has been identified and classified. So the "trigger" is a feature of the argument relation.

- ENTITY DISTANCE: the distance between the two candidate arguments in the sentence, namely the number of intervening words

- Whether the two candidate arguments occur on the same side of the trigger

- PARENT DEPENDENCY DISTANCE: the distance between the two candidate arguments' parents in the dependency parse tree, namely, the path length.

- PARENT POS: if the two candidate arguments share the same parent, take the common parent's POS tag as a feature

- Whether the two candidate arguments occur on the same side of the common parent if the two candidate arguments share the same parent

For an entity pair, if both of the entities belong to the same event's arguments, we take it as positive example. For each positive example, we randomly exchange one of the entities with an irrelevant entity (an irrelevant entity is in the same sentence as the event, but it is not the event's argument) to get a negative example. In the testing procedure, we predict the relationship between entity i and entity j using the maximum entropy classifier.

When the output of the maximum entropy classifier is around 0.5, it is not easy to figure out whether it is the first relation or the second. We call this kind of information "uncertain information"(unclear correlation). For better performance, we **strengthen** the certain information and weaken the uncertain information. We set two thresholds, if the output of the maximum entropy classifier is larger than 0.8, we set $C_{i,j} = 1$ (positive correlation), if the output is lower than 0.2, we set $C_{i,j} = -1$ (negative correlation), otherwise, we set $C_{i,j} = 0$ (unclear correlation). The strengthen mapping is similar to the hard tanh in neural network. If we do not do this, according to the experiment, the performance cannot beat most of the baselines since the uncertain information has very bad noise.

6 Experiments

6.1 Data

We utilize ACE 2005 data sets as our testbed. As is consistent with previous work, we randomly select 10 newswire texts from ACE 2005 training corpora as our development set, and then conduct blind test on a separate set of 40 ACE 2005 newswire texts. The remaining 529 documents in ACE training corpus are used as the training data.

The training dataset of the argument relationship matrix contains 5826 cases (2904 positive and 2922 negative) which are randomly generated according to the ground truth in the 529 training documents.

6.2 Systems to Compare

We compare our system against the following systems:

- **JET** is the baseline of (Grishman et al., 2005), we report the paper values of this method;

- **Cross-Document** is the method proposed by Ji and Grishman (2008), which uses topic-related documents to help extract events in the current document;

- **Cross-Event** is the method proposed by Liao and Grishman (2010), which uses document-level information to improve the performance of ACE event extraction.

- **Cross-Entity** is the method proposed by Hong et al. (2011), which extracts events using cross-entity inference.

- **Joint** is the method proposed by Li et al. (2013), which extracts events based on structure prediction. It is the best-reported structure-based system.

- **DMCNN** is the method proposed by Chen et al. (2015), which uses a dynamic multi-pooling convolutional neural network to extract events. It is the only neural network based method.

The Cross-Document, Cross-Event and Cross-Entity are all extensions of JET. Among these

Method	Trigger Classification			Argument Identification			Argument Role		
	P	R	F_1	P	R	F_1	P	R	F_1
JET	67.6	53.5	59.7	46.5	37.2	41.3	41.0	32.8	36.5
Cross-Event	68.7	**68.9**	68.8	50.9	49.7	50.3	45.1	44.1	44.6
Cross-Entity	72.9	64.3	68.3	53.4	52.9	53.1	51.6	45.5	48.3
Joint	73.7	62.3	67.5	**69.8**	47.9	56.8	**64.7**	44.4	52.7
DMCNN	**75.6**	63.6	**69.1**	68.8	51.9	59.1	62.2	46.9	53.5
RBPB(JET)	62.3	59.9	61.1	50.4	45.8	48.0	41.9	36.5	39.0
+ ET	66.7	65.9	66.3	60.6	56.7	58.6	49.2	48.3	48.7
+ Regu	67.2	61.7	64.3	62.8	57.5	60.0	52.6	48.4	50.4
+ ET + Regu	70.3	67.5	68.9	63.2	**59.4**	**61.2**	54.1	**53.5**	**53.8**

Table 1: Overall performance with gold-standard entities, timex, and values, the candidate arguments are annotated in ACE 2005. "ET" means the pattern balancing event type classifier, "Regu" means the regularization method

methods, Cross-Event, Cross-Entity, and DMCNN make use of the gold-standard entities, timex, and values annotated in the corpus as the argument candidates. Cross-Document uses the JET system to extract candidate arguments. Li et al. (2013) report the performance with both gold-standard argument candidates and predicted argument candidates. Therefore, we compare our results with methods based on gold argument candidates in Table 1 and methods based on predicted argument candidates in Table 2.

We have done a series of ablation experiments:

- **RBPB(JET)**: Our own implementation of JET
- **RBPB(JET) + ET**: Add pattern balanced event type classifier to RBPB(JET)
- **RBPB(JET) + Regu**: Add regularization mechanism to RBPB(JET)
- **RBPB(JET) + ET + Regu**: Add both pattern balanced event type classifier and regularization mechanism to RBPB(JET)

6.2.1 The Selection of Hyper-parameters

We tune the coefficients λ_1 and λ_2 of Eq 2 on the development set, and finally we set $\lambda_1 = 0.10$ and $\lambda_2 = 0.45$. Figure 3 shows the variation of argument identification's F_1 measure and argument classification's F_1 measure when we fix one parameter and change another. Note that the third coefficient $1 - \lambda_1 - \lambda_2$ must be positive, which is the reason why the curve decreases sharply when λ_2 is fixed and $\lambda_1 > 0.65$. Therefore, Figure 3 illustrates that the robustness of our method is very good, which means if the hyperparameters λ_1, λ_2 are larger or smaller, it will not affect the result very much.

6.3 Experiment Results

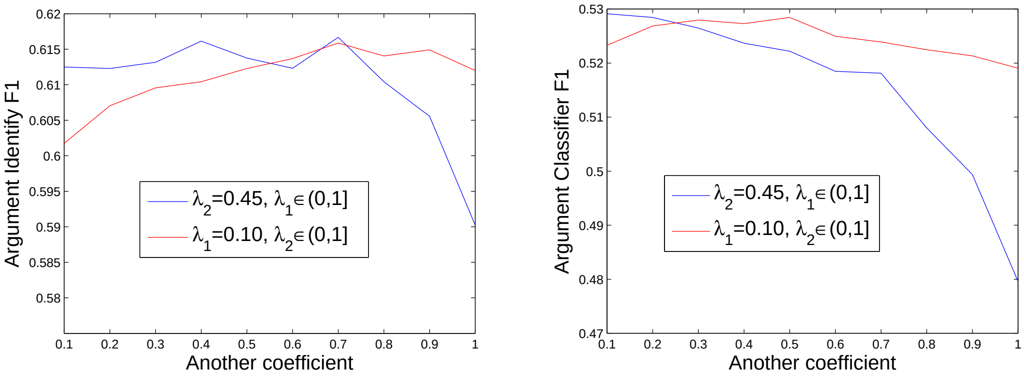
We conduct experiments to answer the following questions. (1) Can pattern balancing lead to a higher performance in trigger classification, argument identification, and classification while retaining the precision value? (2) Can the regularization step improve the performance of argument identification and classification?

Table 1 shows the overall performance on the blind test set. We compare our results with the JET baseline as well as the Cross-Event, Cross-Entity, and joint methods. When adding the event type classifier, in the line titled "+ ET", we see a significant increase in the three measures over the JET baseline in recall. Although our trigger's precision is lower than RBPB(JET), it gains 5.2% improvement on the trigger's F_1 measure, 10.6% improvement on argument identification's F_1 measure and 9.7% improvement on argument classification's F_1 measure. We also test the performance with argument candidates automatically extracted by JET in Table 2, our approach "+ ET" again significantly outperforms the JET baseline. Remarkably, our result is comparable with the Joint model although we only use lexical features.

The line titled "+ Regu" in Table 1 and Table 2 represents the performance when we only use the regularization method. In Table 1, Compared to the four baseline systems, the argument identifi-

Method	Trigger F_1	Arg id F_1	Arg id+cl F_1
JET	59.7	42.5	36.6
Cross-Document	67.3	46.2	42.6
Joint	65.6	-	41.8
RBPB(JET)	60.4	44.3	37.1
+ ET	66.0	47.8	39.7
+ Regu	64.8	54.6	42.0
+ ET + Regu	**67.8**	**55.4**	**43.8**

Table 2: Overall performance with predicted entities, timex, and values, the candidate arguments are extracted by JET. "ET" is the pattern balancing event type classifier, "Regu" is the regularization method



(a) Arg Identify

(b) Arg Classify

Figure 3: The trend graph when fix one coefficient and change another

cation's F_1 measure of "+ Regu" is significantly higher. In Table 2, the "+ Regu" again gains a higher F_1 measure than the JET, Cross-Document, joint model baseline and "+ ET".

The complete approach is denoted as "RBPB" in Table 1 and Table 2. Remarkably, our approach performances comparable in trigger classification with the state-of art methods: Cross-Document, Cross-Event, Cross-Entity, Joint model, DMCNN and significantly higher than them in argument identification as well as classification although we did not use the cross-document, cross-event information or any global feature. Therefore, the relationship between argument candidates can indeed contribute to argument identification performance. The event type classifier also contributes a lot in trigger identification & classification. We do the Wilcoxon Signed Rank Test on trigger classification, argument identification and argument classification, all the three have $p < 0.01$.

A more detailed study of the pattern feature's effect is shown in Table 3. We can see that RBPB with both plain feature and pattern feature can gain much better performance than with two kinds of features alone.

Method (RBPB)	Trigger	Arg id	Arg id+cl
+ Plain feature	66.0	60.5	50.4
+ Pattern feature	65.8	60.1	49.2
+ Both	**68.9**	**61.2**	**53.8**

Table 3: The effect (F_1 value) of pattern feature

However, our approach is just a pipeline approach which suffers from error propagation and the argument performance may not affect the trigger too much. We can see from Table 1 that although we use gold argument candidates, the trigger performance is still lower than DMCNN. Another limitation is that our regularization method does not improve the argument classification too much since it only uses constraints to affect roles. Future work may be done to solve these two limitations.

6.4 Analysis of Argument Relationships

The accuracy of the argument relationship max-ent classifier is 82.4%. Fig 4 shows an example of the argument relationship matrix, which works

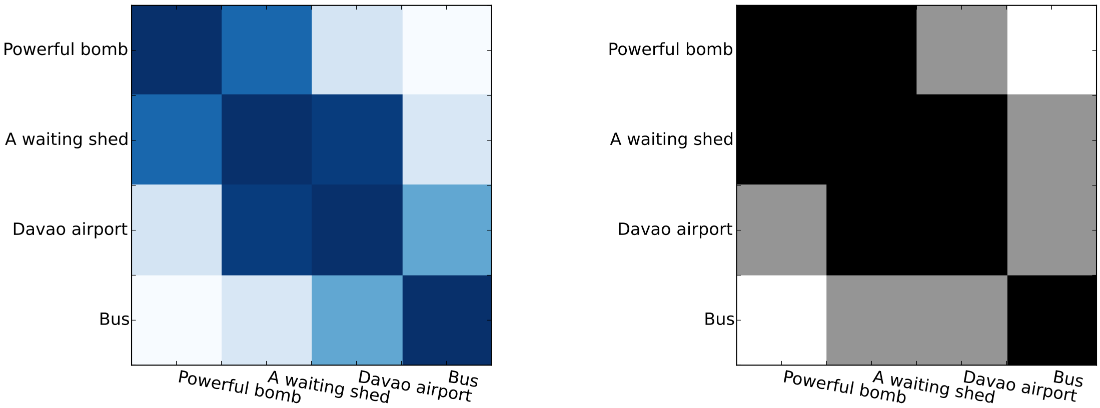


Figure 4: The Argument Relationship Matrix. Left is the origin matrix. Right is the strengthened matrix

for the sentence in Fig 1. In the left part of Fig 4, we can see the argument relationship we capture directly (the darker blue means stronger connection, lighter blue means weaker connection). After strengthening, on the right, the entities with strong connections are classified as positive correlations (the black squares), weak connections are classified as negative correlations (the white squares). Others (the grey squares) are unclear correlations. We can see that positive correlation is between “Powerful bomb” and “A waiting shed” as well as “A waiting shed” and “Davao airport”. Therefore, these entities tend to be extracted at the same time. However, “Powerful bomb” and “Bus” has a negative correlation, so they tend not to be extracted at the same time. In practice, the argument probability of “Powerful bomb” and “A waiting shed” are much higher than the other two. Therefore, “Powerful bomb”, “A waiting shed” and “Davao airport” are the final extraction results.

7 Conclusion

In this paper, we propose two improvements based on the event extraction baseline JET. We find that JET depends too much on event patterns for event type priori and JET considers each candidate argument separately. However, patterns cannot cover all events and the relationship between candidate arguments may help when identifying arguments. For a trigger, if no pattern can be matched, the event type cannot be assigned and the arguments cannot be correctly identified and classified. Therefore, we develop an event type classifier to assign the event type, using both pattern matching information and other features, which gives our system the capability to deal with failed match cases when using patterns alone.

On the other hand, we train a maximum entropy classifier to predict the relationship between candidate arguments. Then we propose a regularization method to make full use of the argument relationship. Our experiment results show that the regularization method is a significant improvement in argument identification over previous works.

In summary, by using the event type classifier and the regularization method, we have achieved a good performance in which the trigger classification is comparable to state-of-the-art methods, and the argument identification & classification performance is significantly better than state-of-the-art methods. However, we only use sentence-level features and our method is a pipelined approach. Also, the argument classification seems not to be affected too much by the regularization. Future work may be done to integrate our method into a joint approach, use some global feature, which may improve our performance. The code is available at `https://github.com/shalei120/RBPB/tree/master/RBET_release`

Acknowledgements

We would like to thank our three anonymous reviewers for their helpful advice on various aspects of this work. This research was supported by the National Key Basic Research Program of China (No.2014CB340504) and the National Natural Science Foundation of China (No.61375074,61273318). The contact author for this paper is Zhifang Sui.

References

Mary Elaine Califf and Raymond J Mooney. 2003. Bottom-up relational learning of pattern matching rules for information extraction. *The Journal of Machine Learning Research*, 4:177–210.

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

Yubo Chen, Liheng Xu, Kang Liu, Daojian Zeng, and Jun Zhao. 2015. Event extraction via dynamic multi-pooling convolutional neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 167–176, Beijing, China, July. Association for Computational Linguistics.

Hai Leong Chieu and Hwee Tou Ng. 2002. A maximum entropy approach to information extraction from semi-structured and free text. *AAAI/IAAI*, 2002:786–791.

Fabio Ciravegna et al. 2001. Adaptive information extraction from text by rule induction and generalisation. In *International Joint Conference on Artificial Intelligence*, volume 17, pages 1251–1256. LAWRENCE ERLBAUM ASSOCIATES LTD.

Aidan Finn and Nicholas Kushmerick. 2004. *Multilevel boundary classification for information extraction*. Springer.

Dayne Freitag. 1998a. Multistrategy learning for information extraction. In *ICML*, pages 161–169.

Dayne Freitag. 1998b. Toward general-purpose learning for information extraction. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics-Volume 1*, pages 404–408. Association for Computational Linguistics.

Ralph Grishman, David Westbrook, and Adam Meyers. 2005. Nyus english ace 2005 system description. *ACE*, 5.

Zhenmei Gu and Nick Cercone. 2006. Segment-based hidden markov models for information extraction. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 481–488. Association for Computational Linguistics.

Yu Hong, Jianfeng Zhang, Bin Ma, Jianmin Yao, Guodong Zhou, and Qiaoming Zhu. 2011. Using cross-entity inference to improve event extraction. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 1127–1136. Association for Computational Linguistics.

Ruihong Huang and Ellen Riloff. 2011. Peeling back the layers: detecting event role fillers in secondary contexts. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 1137–1147. Association for Computational Linguistics.

Ruihong Huang and Ellen Riloff. 2012. Bootstrapped training of event extraction classifiers. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 286–295. Association for Computational Linguistics.

Scott B Huffman. 1996. Learning information extraction patterns from examples. In *Connectionist, Statistical and Symbolic Approaches to Learning for Natural Language Processing*, pages 246–260. Springer.

Heng Ji and Ralph Grishman. 2008. Refining event extraction through cross-document inference. In *Proceedings of the 46st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 254–262.

Jun-Tae Kim and Dan I Moldovan. 1993. Acquisition of semantic patterns for information extraction from corpora. In *Artificial Intelligence for Applications, 1993. Proceedings., Ninth Conference on*, pages 171–176. IEEE.

Yaoyong Li, Kalina Bontcheva, and Hamish Cunningham. 2005. Using uneven margins svm and perceptron for information extraction. In *Proceedings of the Ninth Conference on Computational Natural Language Learning*, pages 72–79. Association for Computational Linguistics.

Qi Li, Heng Ji, and Liang Huang. 2013. Joint event extraction via structured prediction with global features. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 73–82, Sofia, Bulgaria, August. Association for Computational Linguistics.

Shasha Liao and Ralph Grishman. 2010. Using document level cross-event inference to improve event extraction. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 789–797. Association for Computational Linguistics.

Ting Liu and Tomek Strzalkowski. 2012. Bootstrapping events and relations from text. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 296–305. Association for Computational Linguistics.

Wei Lu and Dan Roth. 2012. Automatic event extraction with structured preference modeling. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 835–844. Association for Computational Linguistics.

Mstislav Maslennikov and Tat-Seng Chua. 2007. A multi-resolution framework for information extraction from free text. In *ANNUAL MEETING-ASSOCIATION FOR COMPUTATIONAL LINGUISTICS*, volume 45, page 592. Citeseer.

Siddharth Patwardhan and Ellen Riloff. 2007. Effective information extraction with semantic affinity patterns and relevant regions. In *EMNLP-CoNLL*, volume 7, pages 717–727. Citeseer.

Siddharth Patwardhan and Ellen Riloff. 2009. A unified model of phrasal and sentential evidence for information extraction. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1*, pages 151–160. Association for Computational Linguistics.

Ellen Riloff et al. 1993. Automatically constructing a dictionary for information extraction tasks. In *AAAI*, pages 811–816.

Ellen Riloff, Rosie Jones, et al. 1999. Learning dictionaries for information extraction by multi-level bootstrapping. In *AAAI/IAAI*, pages 474–479.

Ellen Riloff. 1996. Automatically generating extraction patterns from untagged text. In *Proceedings of the national conference on artificial intelligence*, pages 1044–1049.

Satoshi Sekine. 2006. On-demand information extraction. In *Proceedings of the COLING/ACL on Main conference poster sessions*, pages 731–738. Association for Computational Linguistics.

Yusuke Shinyama and Satoshi Sekine. 2006. Preemptive information extraction using unrestricted relation discovery. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 304–311. Association for Computational Linguistics.

Stephen Soderland, David Fisher, Jonathan Aseltine, and Wendy Lehnert. 1995. Crystal: Inducing a conceptual dictionary. *arXiv preprint cmp-lg/9505020*.

Mark Stevenson and Mark A Greenwood. 2005. A semantic approach to ie pattern induction. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 379–386. Association for Computational Linguistics.

Kiyoshi Sudo, Satoshi Sekine, and Ralph Grishman. 2003. An improved extraction pattern representation model for automatic ie pattern acquisition. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pages 224–231. Association for Computational Linguistics.

Roman Yangarber, Ralph Grishman, Pasi Tapanainen, and Silja Huttunen. 2000. Automatic acquisition of domain knowledge for information extraction. In *Proceedings of the 18th conference on Computational linguistics-Volume 2*, pages 940–946. Association for Computational Linguistics.

Kun Yu, Gang Guan, and Ming Zhou. 2005. Resume information extraction with cascaded hybrid model. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 499–506. Association for Computational Linguistics.