Joint Event Trigger Identification and Event Coreference Resolution with Structured Perceptron

Jun Araki and **Teruko Mitamura**
Language Technologies Institute
Carnegie Mellon University
Pittsburgh, PA 15213, USA
{junaraki,teruko}@cs.cmu.edu

Abstract

Events and their coreference offer useful semantic and discourse resources. We show that the semantic and discourse aspects of events interact with each other. However, traditional approaches addressed event extraction and event coreference resolution either separately or sequentially, which limits their interactions. This paper proposes a document-level structured learning model that simultaneously identifies event triggers and resolves event coreference. We demonstrate that the joint model outperforms a pipelined model by 6.9 BLANC F1 and 1.8 CoNLL F1 points in event coreference resolution using a corpus in the biology domain.

1 Introduction

Events convey semantic information such as who did what to whom where and when. They also corefer to each other, playing a role of discourse connection points to form a coherent story. These aspects of events have been already utilized in a wide variety of natural language processing (NLP) applications, such as automated population of knowledge bases (Ji and Grishman, 2011), topic detection and tracking (Allan, 2002), question answering (Bikel and Castelli, 2008), text summarization (Li et al., 2006), and contradiction detection (de Marneffe et al., 2008). This fact illustrates the importance of event extraction and event coreference resolution.

Those semantic and discourse aspects of events are not independent from each other, and in fact often work in interactive manners. We give two examples of the interactions:

(1) British bank Barclays had agreed to **buy**(E1) Spanish rival Banco Zaragozano for 1.14 billion euros. The **combination**(E2) of the banking operations of Barclays Spain and Zaragozano will bring together two complementary businesses.

(2) The Palestinian Authority condemned the **attack**(E3), saying **it**(E4) would divert international sympathy away from the far higher Palestinian civilian death toll.

E1 corefers to E2, and E3 does to E4. E2 is more abstract than E1, and has less evidence of being an event. E4 is a pronoun, and thus may seem to refer to an entity rather than an event. Thus, E2 and E4 are relatively difficult to be recognized as events by themselves. However, event coreference E1-E2, which is supported primarily by E2's participants *Barclays* and *Zaragozano* shared with E1, helps determine that E2 is an event. The same logic applies to E3 and E4. On the other hand, previous works typically rely on a pipelined model that extracts events (e.g., E1 and E3) at the first stage, and then resolves event coreference at the second stage. Although this modularity is preferable from development perspectives, the pipelined model limits the interactions. That is, the first stage alone is unlikely to detect E2 and E4 as events due to the difficulties described above. These missing events make it impossible for the second stage to resolve event coreference E1-E2 and E3-E4.

In this work, we address the problem using the ProcessBank corpus (Berant et al., 2014). Following the terminology defined in the corpus, we introduce several terms:

- **Event**: an abstract representation of a change of state, independent from particular texts.
- **Event trigger**: main word(s) in text, typically a verb or a noun that most clearly expresses an event.
- **Event arguments**: participants or attributes in text, typically nouns, that are involved in an event.
- **Event mention**: a clause in text that describes an event, and includes both a trigger and arguments.
- **Event coreference**: a linguistic phenomenon that two event mentions refer to the same event.

We aim to explore the interactions between event mentions and event coreference. As a first step toward the goal, we focus on the task of identifying event triggers and resolving event coreference, and

Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pages 2074–2080,
Lisbon, Portugal, 17-21 September 2015. ©2015 Association for Computational Linguistics.

propose a document-level joint learning model using structured perceptron (Collins, 2002) that simultaneously predicts them. Our assumption is that the joint model is able to capture the interactions between event triggers and event coreference adequately, and such comprehensive decision improves the system performance. For instance, the joint model is likely to extract E2 as well as E1 successfully via their event coreference by simultaneously looking at coreference features.

Our contributions are as follows:

1. This is the first work that simultaneously predicts event triggers and event coreference using a single joint model. At the core of the model is a document-level structured perceptron algorithm that learns event triggers and event coreference jointly.
2. The incremental token-based prediction in joint decoding poses a challenge of synchronizing the assignments of event triggers and coreference. To avoid this problem, we propose an incremental decoding algorithm that combines the segment-based decoding and best-first clustering algorithm.
3. Our experiments indicate that the joint model achieves a substantial performance gain in event coreference resolution with a corpus in the biology domain, as compared to a pipelined model.

2 Related Work

No previous work deals with event extraction and event coreference resolution simultaneously. We thus describe how these two tasks have been addressed separately, and how joint structured learning has been studied in other NLP tasks.

Event extraction has been studied mainly in the newswire domain and the biomedical domain as the task of detecting event triggers and determining their event types and arguments. In the former domain, most work took a pipelined approach where local classifiers identify triggers first, and then detect arguments (Ji and Grishman, 2008; Liao and Grishman, 2010; Hong et al., 2011). Li et al. (2013) presented a structured perceptron model to detect triggers and arguments jointly. Similarly, joint dependencies in events were also addressed in the latter domain (Poon and Vanderwende, 2010; McClosky et al., 2011; Riedel and McCallum, 2011; Venugopal et al., 2014). However, none of them incorporated event coreference into their model.

Event coreference resolution is more challenging and less explored. To set up event triggers as a starting point of the task, some works use human annotation in a corpus (Bejan and Harabagiu, 2014; Liu et al., 2014), and others use the output of a separate event extraction system (Lee et al., 2012). Berant et al. (2014) presented a model that jointly predicts event arguments and event coreference (as well as other relations between event triggers). However, none of them tries to predict event triggers and event coreference jointly.

Joint structured learning has been applied to several NLP tasks, such as word segmentation and part-of-speech (POS) tagging (Zhang and Clark, 2008a), POS tagging and dependency parsing (Bohnet and Nivre, 2012), dependency parsing and semantic role labeling (Johansson and Nugues, 2008), the extraction of event triggers and arguments (Li et al., 2013), and the extraction of entity mentions and relations (Li and Ji, 2014). Their underlying ideas are similar to ours. That is, one can train a structured learning model to globally capture the interactions between two relevant tasks via a certain kind of structure, while making predictions specifically for these respective tasks. However, no prior work has studied the interactions between event trigger identification and event coreference resolution.

3 Approach

We formalize the extraction of event triggers and event coreference as a problem of structured prediction. The output structure is a document-level event graph where each node represents an event trigger, and each edge represents an event coreference link between two event triggers.

3.1 Corpus

The ProcessBank corpus consists of 200 paragraphs from the textbook *Biology* (Campbell and Reece, 2005). Table 1 shows statistics of our data splits. The original corpus provides 150 paragraphs as training data, and we split them into 120 and 30 for our training and development, respectively. We chose ProcessBank instead of a larger corpus such as the Automatic Content Extraction (ACE) 2005 corpus for the following two reasons. First, the human annotation of event coreference links in ProcessBank enables us to apply the best-first clustering directly; on the other hand, this is not readily feasible in ACE 2005 since it annotates event coreference as clusters, and gold standard event coreference links required for the best-first clustering are not available. Second, event coreference resolution using ProcessBank is novel since almost no previous work on the task used that corpus. The only exception could be (Berant et al., 2014), where they extracted several types of relations between event triggers, including event coreference. However, they did not report any performance scores of their system specifically on event coreference, and thus their work is not comparable to ours.

	Train	Dev	Test	Total
# of paragraphs	120	30	50	200
# of event triggers	823	224	356	1403
# of event coreferences	73	28	30	131

Table 1: Statistics of our dataset.

Unlike previous work (Berant et al., 2014; Li et al., 2013), we explicitly allow an event trigger to have multiple tokens, such as verb phrase 'look into' and compound proper noun 'World War II'. This is a more realistic setting for event trigger identification since in general there are a considerable number of multi-token event triggers[1].

3.2 Event Graph Learning

Let x denote an input document with n tokens where x_i is the i-th token in the document. For event graph learning, we use structured perceptron (Collins, 2002), and average weights to reduce overfitting as suggested in (Collins, 2002). The algorithm involves decoding to generate the best event graph for each input document. We elaborate on our decoding algorithm in Section 3.3. Since an event graph has an exponentially large search space, we use beam search to approximate exact inference. We extract a range of features by using Stanford CoreNLP (Manning et al., 2014), MATE (Björkelund et al., 2009), OpenNLP[2], Nomlex (Macleod et al., 1998), and Levin verb classes (Levin, 1993). For brevity, we provide details of the structured perceptron algorithm and features in the supplementary material.

We use the standard-update strategy in our structured perceptron model. As variants of structured perceptron, one could employ the early update (Collins and Roark, 2004) and max-violation update (Huang et al., 2012) to our model. Our initial experiments indicated that early updates happen too early to gain sufficient feedback on weights from entire documents in training examples, ending up with a poorer performance than the standard update. This contrasts with the fact that the early-update strategy was successfully applied to other NLP tasks such as constituent parsing (Collins and Roark, 2004) and dependency parsing (Zhang and Clark, 2008b). The main reason why the early update fell short of the standard update in our setting is that joint event trigger identification and event coreference resolution is a much more difficult task since they require more complex knowledge and argument structures. Due to the difficultly of the task, it is also very difficult to develop such an effective feature set that beam search can explore the search space of an entire document thoroughly with early updates. This observation follows (Björkelund and Kuhn, 2014) on entity coreference resolution. In contrast, the max-violation update showed almost the same performance as the standard update on the development data. From these results, we chose the standard-update strategy for simplicity.

3.3 Joint Decoding

Given that an event trigger has one or more tokens, event trigger identification could be solved as a token-level sequential labeling problem with BIO or BILOU scheme in the same way as named entity recognition (Ratinov and Roth, 2009). If one uses this approach, a beam state may represent a partial assignment of an event trigger. However, event coreference can be explored only from complete assignments of an event trigger. Thus, one would need to synchronize the search process of event coreference by comparing event coreferences from the complete assignment at a certain position with those from complete assignments at following positions. This makes it complicated to implement the formalization of token-level sequential labeling for joint decoding in our task. One possible way to avoid this problem is to extract event trigger candidates with a preference on high recall first, and then search event coreference from those candidates, regarding them as complete assignments of an event trigger. This recall-oriented pre-filtering is often used in entity coreference resolution (Lee et al., 2013; Björkelund

[1] For example, around 13.4% of the 1403 event triggers in ProcessBank have multiple tokens.

[2] http://opennlp.apache.org/

Algorithm 1 Joint decoding for event triggers and coreference with beam search.

```
Input:  input document $x = (x_1, x_2, \dots, x_n)$
Input:  beam width $k$, max length of event trigger $l_{max}$
Output: best event graph $\hat{y}$ for $x$
 1: initialize empty beam history $B[1..n]$
 2: for $i \leftarrow 1..n$ do
 3:     for $l \leftarrow 1..l_{max}$ do
 4:         for $y \in B[i-l]$ do
 5:             $e \leftarrow$ CreateEventTrigger$(l, i)$.
 6:             AppendEventTrigger$(y, e)$
 7:             $B[i] \leftarrow k$-Best$(B[i] \cup y)$
 8:             for $j \leftarrow 1..i-1$ do
 9:                 $c \leftarrow$ CreateEventCoref$(j, e)$.
10:                 AddEventCoref$(y, c)$
11:                 $B[i] \leftarrow k$-Best$(B[i] \cup y)$
12: return $B[n][0]$
```

and Farkas, 2012). In our initial experiments, we observed that our rule-based filter gained around 97% recall, but extracted around 12,400 false positives against 823 true positives in the training data. This made it difficult for our structured perceptron to learn event triggers, which underperformed on event coreference resolution.

We, therefore, employ segment-based decoding with multiple-beam search (Zhang and Clark, 2008a; Li and Ji, 2014) for event trigger identification, and combine it with the best-first clustering (Ng and Cardie, 2002) for event coreference resolution in document-level joint decoding. The key idea of segment-based decoding with multiple-beam search is to keep previous beam states available, and use them to form segments from previous positions to the current position. Let l_{max} denote the upper bound on the number of tokens in one event trigger. The k-best partial structures (event subgraphs) in beam B at the j-th token is computed as follows:

$$B[j] = \underset{y \in \{y_{[1:j-l]} \in B[j-l],\ y_{[j-l+1,j]} = s\}}{k\text{-Best}} \Phi(x, y) \cdot \mathbf{w}$$

where $1 \leq l \leq l_{max}$, $y_{[1:j]}$ is an event subgraph ending at the j-th token, and $y_{[j-l+1,j]} = s$ means that partial structure $y_{[j-l+1,j]}$ is a segment, i.e., an event trigger candidate with a subsequence of tokens $x_{[j-l+1,j]}$. This approximates Viterbi decoding with beam search.

The best-first clustering incrementally makes coreference decisions by selecting the most likely antecedent for each trigger. Our joint decoding algorithm makes use of the incremental process to combine the segment-based decoding and best-first clustering. Algorithm 1 shows the summary of the joint decoding algorithm. Line 3 - 7 implements the segment-based decoding, and line 8 - 11 implements the best-first clustering. Once a new event trigger is appended to an event subgraph at line 6, the decoder uses it as a referring mention regardless of whether the event subgraph is in the beam, and seeks the best antecedent for it. This enables the joint model to make a more global decision on event trigger identification and event coreference decision, as described in Section 1.

4 Experimental Settings

When training our model, we observed that 20-iteration training almost reached convergence, and thus we set the number of iterations to 20. We set l_{max} to 6 because we observed that the longest event trigger in the entire ProcessBank corpus has six tokens. When tuning beam width k on the development set, large beam width did not give us a significant performance difference. We attribute this result to the small size of the development data. In particular, the development data has only 28 event coreferences, which makes it difficult to reveal the effect of beam width. We thus set k to 1 in our experiments.

4.1 Baseline Systems

Our baseline is a pipelined model that divides the event trigger decoding and event coreference decoding in Algorithm 1 into two separate stages. It uses the same structured perceptron with the same hyperparameters and feature templates. We choose this baseline because it clearly reveals the effectiveness of the joint model by focusing only on the architectural difference. One could develop other baseline systems. One of them is a deterministic sieve-based approach by Lee et al. (2013). A natural extension to the approach for performing event trigger identification as well as event coreference resolution would be to develop additional sieves to classify singletons into real event triggers or spurious ones. We leave it for future work.

4.2 Evaluation

We evaluate our system using a reference implementation of coreference scoring algorithms (Pradhan et al., 2014; Luo et al., 2014). As for event trigger identification, this scorer computes precision (P), recall (R), and the F1 score. With respect to event coreference resolution, the scorer computes MUC (Vilain et al., 1995), B^3 (Bagga and Baldwin, 1998), two CEAF metrics CEAF$_m$ and CEAF$_e$ (Luo, 2005), and

	MUC			B^3			CEAF$_m$			CEAF$_e$			BLANC			CoNLL
System	R	P	F1	R	P	F1	R	P	F1	R	P	F1	R	P	F1	F1
Baseline	26.66	19.51	22.53	55.47	58.64	57.01	53.08	60.38	56.50	52.68	63.14	57.44	30.13	25.10	25.05	45.66
Joint	20.00	37.50	26.08	53.37	63.36	57.93	53.93	62.95	58.09	55.06	62.11	58.38	27.51	38.43	31.91	47.45

Table 2: Results of event coreference resolution. 'Baseline' refers to the second stage of our baseline.

BLANC (Recasens and Hovy, 2011) extended by Luo et al. (2014). We also report the CoNLL average (Denis and Baldridge, 2009), which is the average of MUC F1, B^3 F1, and CEAF$_e$ F1.

5 Results and Discussions

We first show the result of event coreference resolution on the test data in Table 2. The joint model outperforms the baseline by 6.9 BLANC F1 and 1.8 CoNLL F1 points. We observed that this overall performance gain comes largely from a precision gain, more specifically, substantially reduced false positives. We explain the superiority of the joint model as follows. In the baseline, the second stage uses the output of the first stage. Since event triggers are fixed at this point, the baseline explores coreference links only between these event triggers. In contrast, the joint model seeks event triggers and event coreference simultaneously, and thus it explores a larger number of false positives in the search process, thereby learning to penalize false positives more adequately than the baseline.

System	Recall	Precision	F1
Baseline	57.02	64.85	60.68
Joint	55.89	65.24	60.21

Table 3: Results of event trigger identification. 'Baseline' refers to the first stage of our baseline.

Table 3 shows the results of event trigger identification on the test data. We observed that the joint model also reduced false positives, similarly in event coreference resolution. However, its improvement on precision is small, ending up with almost the same F1 point as the baseline. We speculate that this is due to the small size of the corpus, and the joint model was unable to show its advantages in event trigger identification.

Below are two error cases in event coreference resolution, where our model fails to resolve E5-E6 and E7-E8. The model was unable to adequately extract features for both event triggers and event coreference, particularly because their surface strings are not present in training data, they are lexically and syntactically different, and they do not share key semantic roles (e.g., agents and patients) in a clear argument structure.

(3) When the cell is stimulated, gated channels open that facilitate Na+ **diffusion**(E5). Sodium ions then **"fall"**(E6) down their electrochemical gradient, . . .

(4) The next seven steps **decompose**(E7) the citrate back to oxaloacetate. It is this **regeneration**(E8) of oxaloacetate that makes this process a cycle.

6 Conclusion and Future Work

We present a joint structured prediction model for event trigger identification and event coreference resolution. To our knowledge, this is the first work that solves these two tasks simultaneously. Our experiment shows that the proposed method effectively penalizes false positives in joint search, thereby outperforming a pipelined model substantially in event coreference resolution.

There are a number of avenues for future work. One can further ensure the advantage of the joint model using a larger corpus. Our preliminary experiment on the ACE 2005 corpus shows that due to its larger document size and event types, one will need to reduce training time by a distributed learning algorithm such as mini-batches (Zhao and Huang, 2013). Another future work is to incorporate other components of events into the model. These include event types, event arguments, and other relations such as subevents. One could leverage them as other learning targets or constraints, and investigate further benefits of joint modeling.

Acknowledgments

This research was supported in part by DARPA grant FA8750-12-2-0342 funded under the Deep Exploration and Filtering of Text (DEFT) Program, and by U.S. Army Research Office (ARO) grant W911NF-14-1-0436 under the Reading, Extraction, and Assembly of Pathways for Evidentiary Reading (REAPER) Program. Any opinion, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the view of DARPA, ARO, or the U.S. government. Jun Araki is partly supported by a Funai Overseas Scholarship.

References

James Allan. 2002. *Topic Detection and Tracking: Event-based Information Organization*. Kluwer Academic Publishers.

Amit Bagga and Breck Baldwin. 1998. Algorithms for scoring coreference chains. In *Proceedings of LREC 1998 Workshop on Linguistics Coreference*, pages 563–566.

Cosmin Adrian Bejan and Sanda M. Harabagiu. 2014. Unsupervised event coreference resolution. *Computational Linguistics*, 40(2):311–347.

Jonathan Berant, Vivek Srikumar, Pei-Chun Chen, Abby Vander Linden, Brittany Harding, Brad Huang, Peter Clark, and Christopher D. Manning. 2014. Modeling biological processes for reading comprehension. In *Proceedings of EMNLP 2014*, pages 1499–1510.

Daniel M. Bikel and Vittorio Castelli. 2008. Event matching using the transitive closure of dependency relations. In *Proceedings of ACL 2008*, pages 145–148.

Anders Björkelund and Richárd Farkas. 2012. Data-driven multilingual coreference resolution using resolver stacking. In *Proceedings of EMNLP/CoNLL 2012*, pages 49–55.

Anders Björkelund and Jonas Kuhn. 2014. Learning structured perceptrons for coreference resolution with latent antecedents and non-local features. In *Proceedings of ACL 2014*, pages 47–57.

Anders Björkelund, Love Hafdell, and Pierre Nugues. 2009. Multilingual semantic role labeling. In *Proceedings of CoNLL 2009*, pages 43–48.

Bernd Bohnet and Joakim Nivre. 2012. A transition-based system for joint part-of-speech tagging and labeled non-projective dependency parsing. In *Proceedings of EMNLP/CoNLL 2012*, pages 1455–1465.

Neil Campbell and Jane Reece. 2005. *Biology*. Benjamin Cummings.

Michael Collins and Brian Roark. 2004. Incremental parsing with the perceptron algorithm. In *Proceedings of ACL 2004*, pages 111–118.

Michael Collins. 2002. Discriminative training methods for Hidden Markov Models: Theory and experiments with perceptron algorithms. In *Proceedings of EMNLP 2002*, pages 1–8.

Marie-Catherine de Marneffe, Anna N. Rafferty, and Christopher D. Manning. 2008. Finding contradictions in text. In *Proceedings of ACL-HLT 2008*, pages 1039–1047.

Pascal Denis and Jason Baldridge. 2009. Global joint models for coreference resolution and named entity classification. *Procesamiento del Lenguaje Natural*, 42:87–96.

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 ACL-HLT 2011*, pages 1127–1136.

Liang Huang, Suphan Fayong, and Yang Guo. 2012. Structured perceptron with inexact search. In *Proceedings of NAACL-HLT 2012*, pages 142–151.

Heng Ji and Ralph Grishman. 2008. Refining event extraction through cross-document inference. In *Proceedings of ACL-HLT 2008*, pages 254–262.

Heng Ji and Ralph Grishman. 2011. Knowledge base population: Successful approaches and challenges. In *Proceedings of ACL-HLT 2011*, pages 1148–1158.

Richard Johansson and Pierre Nugues. 2008. Dependency-based semantic role labeling of PropBank. In *Proceedings of EMNLP 2008*, pages 69–78.

Heeyoung Lee, Marta Recasens, Angel Chang, Mihai Surdeanu, and Dan Jurafsky. 2012. Joint entity and event coreference resolution across documents. In *Proceedings of EMNLP/CoNLL 2012*, pages 489–500.

Heeyoung Lee, Angel Chang, Yves Peirsman, Nathanael Chambers, Mihai Surdeanu, and Dan Jurafsky. 2013. Deterministic coreference resolution based on entity-centric, precision-ranked rules. *Computational Linguistics*, 39(4):885–916.

Beth Levin. 1993. *English Verb Classes and Alternation: A Preliminary Investigation*. The University of Chicago Press.

Qi Li and Heng Ji. 2014. Incremental joint extraction of entity mentions and relations. In *Proceedings of ACL 2014*, pages 402–412.

Wenjie Li, Mingli Wu, Qin Lu, Wei Xu, and Chunfa Yuan. 2006. Extractive summarization using inter- and intra- event relevance. In *Proceedings of ACL/COLING 2006*, pages 369–376.

Qi Li, Heng Ji, and Liang Huang. 2013. Joint event extraction via structured prediction with global features. In *Proceedings of ACL 2013*, pages 73–82.

Shasha Liao and Ralph Grishman. 2010. Using document level cross-event inference to improve event extraction. In *Proceedings of ACL 2010*, pages 789–797.

Zhengzhong Liu, Jun Araki, Eduard Hovy, and Teruko Mitamura. 2014. Supervised within-document event coreference using information propagation. In *Proceedings of LREC 2014*.

Xiaoqiang Luo, Sameer Pradhan, Marta Recasens, and Eduard Hovy. 2014. An extension of BLANC to system mentions. In *Proceedings of ACL 2014*, pages 24–29.

Xiaoqiang Luo. 2005. On coreference resolution performance metrics. In *Proceedings of HLT/EMNLP 2005*, pages 25–32.

Catherine Macleod, Ralph Grishman, Adam Meyers, Leslie Barrett, and Ruth Reeves. 1998. Nomlex: A lexicon of nominalizations. In *Proceedings of EURALEX 1998*, pages 187–193.

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings ACL 2014: System Demonstrations*, pages 55–60.

David McClosky, Mihai Surdeanu, and Christopher Manning. 2011. Event extraction as dependency parsing. In *Proceedings of ACL-HLT 2011*, pages 1626–1635.

Vincent Ng and Claire Cardie. 2002. Improving machine learning approaches to coreference resolution. In *Proceedings of ACL 2002*, pages 104–111.

Hoifung Poon and Lucy Vanderwende. 2010. Joint inference for knowledge extraction from biomedical literature. In *Proceedings of NAACL-HLT 2010*, pages 813–821.

Sameer Pradhan, Xiaoqiang Luo, Marta Recasens, Eduard Hovy, Vincent Ng, and Michael Strube. 2014. Scoring coreference partitions of predicted mentions: A reference implementation. In *Proceedings of ACL 2014*, pages 30–35.

Lev Ratinov and Dan Roth. 2009. Design challenges and misconceptions in named entity recognition. In *Proceedings of CoNLL 2009*, pages 147–155.

Marta Recasens and Eduard Hovy. 2011. BLANC: Implementing the Rand index for coreference evaluation. *Natural Language Engineering*, 17(4):485–510.

Sebastian Riedel and Andrew McCallum. 2011. Fast and robust joint models for biomedical event extraction. In *Proceedings of EMNLP 2011*, pages 1–12.

Deepak Venugopal, Chen Chen, Vibhav Gogate, and Vincent Ng. 2014. Relieving the computational bottleneck: Joint inference for event extraction with high-dimensional features. In *Proceedings of EMNLP 2014*, pages 831–843.

Marc Vilain, John Burger, John Aberdeen, Dennis Connolly, and Lynette Hirschman. 1995. A model-theoretic coreference scoring scheme. In *Proceedings of MUC-6*, pages 45–52.

Yue Zhang and Stephen Clark. 2008a. Joint word segmentation and POS tagging using a single perceptron. In *Proceedings of ACL-HLT 2008*, pages 888–896.

Yue Zhang and Stephen Clark. 2008b. A tale of two parsers: Investigating and combining graph-based and transition-based dependency parsing. In *Proceedings of EMNLP 2008*, pages 562–571.

Kai Zhao and Liang Huang. 2013. Minibatch and parallelization for online large margin structured learning. In *Proceedings of NAACL-HLT 2013*, pages 370–379.

Supplemetary Material: Joint Event Trigger Identification and Event Coreference Resolution with Structured Perceptron

Jun Araki and **Teruko Mitamura**
Language Technologies Institute
Carnegie Mellon University
Pittsburgh, PA 15213, USA
{junaraki,teruko}@cs.cmu.edu

1 Structured Perceptron

We show the structured perceptron algorithm in Algorithm A. In our event graph learning, a training example is represented as a pair of input document x and its associated event graph y. Line 4 involves decoding to generate the best event graph for x. $\Phi(x, y)$ denotes a feature vector function that computes a feature vector for event graph y over x.

Algorithm A Structured perceptron.

```
Input: training examples $\{(x^{(i)}, y^{(i)})\}_{i=1}^{N}$
Input: number of iterations $T$
Output: weight vector $\mathbf{w}$
1: $\mathbf{w} \leftarrow \mathbf{0}$                                ▷ Initialization.
2: for $t \leftarrow 1..T$ do
3:     for $i \leftarrow 1..N$ do
4:         $\hat{y}^{(i)} = \text{argmax}_{y \in \mathcal{Y}(x^{(i)})} \mathbf{w} \cdot \Phi(x^{(i)}, y)$
5:         if $\hat{y}^{(i)} \neq y^{(i)}$ then
6:             $\mathbf{w} \leftarrow \mathbf{w} + \Phi(x^{(i)}, y^{(i)}) - \Phi(x^{(i)}, \hat{y}^{(i)})$
7: return $\mathbf{w}$
```

2 Features for Event Trigger Identification

We use the following features for event trigger identification:

1. Whether to end with a suffix '-tion' or '-ment'
2. Whether to begin with a determiner
3. The number of tokens in the trigger
4. Lemma of the head word
5. Lower-case string of the trigger
6. Part-of-speech tag of the head word
7. All part-of-speech tags of the trigger
8. Relation type of dependency heads
9. Relation type of dependency children
10. Semantic role types if the trigger is a predicate
11. Levin verb classes of the head word
12. Whether the trigger is a title of pages under category 'Biological processes' and related categories

3 Features for Event Coreference Resolution

For the best-first clustering in event coreference resolution, we use the following pairwise features given two event triggers:

1. Whether 3-character prefixes match
2. Whether Lower-case strings match
3. Whether Head word strings match
4. Whether Lemma of head words match
5. Whether the Nomlex nominalization of head words match
6. Whether the following trigger is an acronym of the preceding trigger
7. Whether the following trigger is in apposition to the preceding trigger
8. Part-of-speeches of head words
9. Dependency path between the triggers
10. Whether the following trigger has a determiner
11. Whether both triggers are predicates
12. Whether both triggers share a semantic role
13. Whether both triggers has an argument of the same string with the same semantic role
14. Levin verb classes which both head words belong to
15. Whether the preceding trigger is the first word in a document