# Joint Learning for Event Coreference Resolution

**Jing Lu** and **Vincent Ng**
Human Language Technology Research Institute
University of Texas at Dallas
Richardson, TX 75083-0688
{ljwinnie,vince}@hlt.utdallas.edu

## Abstract

While joint models have been developed for many NLP tasks, the vast majority of event coreference resolvers, including the top-performing resolvers competing in the recent TAC KBP 2016 Event Nugget Detection and Coreference task, are pipeline-based, where the propagation of errors from the trigger detection component to the event coreference component is a major performance limiting factor. To address this problem, we propose a model for jointly learning event coreference, trigger detection, and event anaphoricity. Our joint model is novel in its choice of tasks and its features for capturing cross-task interactions. To our knowledge, this is the first attempt to train a mention-ranking model and employ event anaphoricity for event coreference. Our model achieves the best results to date on the KBP 2016 English and Chinese datasets.

## 1 Introduction

Within-document event coreference resolution is the task of determining which event mentions in a text refer to the same real-world event. Compared to entity coreference resolution, event coreference resolution is not only much less studied, but it is arguably more challenging. The challenge stems in part from the fact that an event coreference resolver typically lies towards the end of the standard information extraction pipeline, assuming as input the noisy outputs of its upstream components. One such component is the trigger detection system, which is responsible for identifying event triggers and determining their event subtypes.

As is commonly known, trigger detection is another challenging task that is far from being solved. In fact, in the recent TAC KBP 2016 Event Nugget Detection and Coreference task, trigger detection (a.k.a. event nugget detection in KBP) is deliberately made more challenging by focusing only on detecting the 18 subtypes of triggers on which the KBP 2015 participating systems' performances were the poorest (Mitamura et al., 2016). The best-performing KBP 2016 system on English trigger detection achieved only an F-score of 47 (Lu and Ng, 2016).[1]

Given the difficulty of trigger detection, it is conceivable that many errors will propagate from the trigger detection component to the event coreference component in any pipeline architecture where trigger detection precedes event coreference resolution. These trigger detection errors could severely harm event coreference performance. For instance, two event mentions could be wrongly posited as coreferent if the underlying triggers were wrongly predicted to have the same subtype. Nevertheless, the top-performing systems in the KBP 2016 event coreference task all adopted the aforementioned pipeline architecture (Liu et al., 2016; Lu and Ng, 2016; Nguyen et al., 2016). Their performances are not particularly impressive, however: the best English event coreference F-score (averaged over four scoring metrics) is only around 30%.

To address this error propagation problem, we describe a joint model of trigger detection, event coreference, and event anaphoricity in this paper. Our choice of these three tasks is motivated in part by their inter-dependencies. As mentioned above, it is well-known that trigger detection performance has a huge impact on event coreference performance. Though largely under-investigated, event coreference could also improve

[1]This is the best English nugget *type* result in KBP 2016. In this paper, we will not be concerned with *realis* classification, as it does not play any role in event coreference.

trigger detection. For instance, if two event mentions are posited as coreferent, then the underlying triggers must have the same event subtype. While the use of anaphoricity information for entity coreference has been extensively studied (see Ng (2010)), to our knowledge there has thus far been no attempt to explicitly model event anaphoricity for event coreference.[2] Although the mention-ranking model we employ for event coreference also allows an event mention to be posited as non-anaphoric (by resolving it to a *null* candidate antecedent), our decision to train a separate anaphoricity model and integrate it into our joint model is motivated in part by the recent successes of Wiseman et al. (2015), who showed that there are benefits in jointly training a noun phrase anaphoricity model and a mention-ranking model for entity coreference resolution. Finally, event anaphoricity and trigger detection can also mutually benefit each other. For instance, any verb posited as a non-trigger cannot be anaphoric, and any verb posited as anaphoric must be a trigger. Note that in our joint model, anaphoricity serves as an *auxiliary* task: its intended use is to improve trigger detection and event coreference, potentially mediating the interaction between trigger detection and event coreference.

Being a structured conditional random field, our model encompasses two types of factors. Unary factors encode the features specific for each task. Binary and ternary factors capture the interaction between each pair of tasks in a soft manner, enabling the learner to learn which combinations of values of the output variables are more probable. For instance, the learner should learn that it is not a good idea to classify a verb both as anaphoric and as a non-trigger. Our model is similar in spirit to Durrett and Klein's (2014) joint model for *entity* analysis, which performs joint learning for entity coreference, entity linking and semantic typing via the use of interaction features.

Our contributions are two-fold. First, we present a joint model of event coreference, trigger detection, and anaphoricity that is novel in terms of the choice of tasks and the features used to capture cross-task interactions. Second, our model achieves the best results to date on the KBP 2016 English and Chinese event coreference tasks.

[2]Following the entity coreference literature, we overload the term *anaphoricity*, saying that an event mention is anaphoric if it is coreferent with a preceding mention in the associated text.

## 2 Definitions, Task, and Corpora

### 2.1 Definitions

We employ the following definitions in our discussion of trigger detection and event coreference:

- An **event mention** is an explicit occurrence of an event consisting of a textual trigger, arguments or participants (if any), and the event type/subtype.
- An **event trigger** is a string of text that most clearly expresses the occurrence of an event, usually a word or a multi-word phrase
- An **event argument** is an argument filler that plays a certain role in an event.
- An **event coreference chain** (a.k.a. an **event hopper**) is a group of event mentions that refer to the same real-world event. They must have the same event (sub)type.

To understand these definitions, consider the example in Table 1, which contains two coreferent event mentions, $ev1$ and $ev2$. *left* is the trigger for $ev1$ and *departed* is the trigger for $ev2$. Both triggers have subtype Movement.Transport-Person. $ev1$ has three arguments, *Georges Cipriani*, *prison*, and *Wednesday* with roles *Person*, *Origin*, and *Time* respectively. $ev2$ also has three arguments, *He*, *Ensisheim*, and *police vehicle* with roles *Person*, *Origin*, and *Instrument* respectively.

### 2.2 Task

The version of the event coreference task we focus on in this paper is the Event Nugget Detection and Coreference task in the TAC KBP 2016 Event Track. While we discuss the role played by event arguments in event coreference in the previous subsection, KBP 2016 addresses event argument detection as a separate shared task. In other words, the KBP 2016 Event Nugget Detection and Coreference task focuses solely on trigger detection and event coreference.

It is worth mentioning that the KBP Event Nugget Detection and Coreference task, which started in 2015, aims to address a major weakness of the ACE 2005 event coreference task. Specifically, ACE 2005 adopts a strict notion of event identity, with which two event mentions were annotated as coreferent if and only if "they had the same agent(s), patient(s), time, and location" (Song et al., 2015), and their event attributes (polarity, modality, genericity, and tense) were not incompatible. In contrast, KBP adopts a more relaxed definition of event coreference, allowing two

| |
|---|
| Georges Cipriani$_{[Person]}$, {left}$_{ev1}$ the prison$_{[Origin]}$ in Ensisheim in northern France on parole on Wednesday$_{[Time]}$. He$_{[Person]}$ {departed}$_{ev2}$ Ensisheim$_{[Origin]}$ in a police vehicle$_{[Instrument]}$ bound for an open prison near Strasbourg. |

Table 1: Event coreference resolution example.

event mentions to be coreferent as long as they *intuitively* refer to the same real-world event. Under this definition, two event mentions can be coreferent even if their time and location arguments are not coreferent. In our example in Table 1, $ev1$ and $ev2$ are coreferent in KBP because they both refer to the same event of Cipriani leaving the prison. However, they are *not* coreferent in ACE because their *Origin* arguments are not coreferent (one Origin argument involves a prison in Ensisheim while the other involves the city Ensisheim).

### 2.3 Corpora

Given our focus on the KBP 2016 Event Nugget Detection and Coreference task, we employ the English and Chinese corpora used in this task for evaluation, referring to these corpora as the KBP 2016 English and Chinese corpora for brevity. There are no official training sets: the task organizers simply made available a number of event coreference-annotated corpora for training. For English, we use LDC2015E29, E68, E73, and E94 for training. These corpora are composed of two types of documents, newswire documents and discussion forum documents. Together they contain 648 documents with 18739 event mentions distributed over 9955 event coreference chains. For Chinese, we use LDC2015E78, E105, and E112 for training. These corpora are composed of discussion forum documents only. Together they contain 383 documents with 4870 event mentions distributed over 3614 event coreference chains.

The test set for English consists of 169 newswire and discussion forum documents with 4155 event mentions distributed over 3191 event coreference chains. The test set for Chinese consists of 167 newswire and discussion forum documents with 2518 event mentions distributed over 1912 event coreference chains. Note that these test sets contain only annotations for event triggers and event coreference (i.e., there are no event argument annotations). While some of the training sets additionally contain event argument annotations, we do not make use of event argument annotations in model training to ensure a fairer comparison to the teams participating in the KBP 2016 Event Nugget Detection and Coreference task.

## 3 Model

### 3.1 Overview

Our model, which is a structured conditional random field, operates at the document level. Specifically, given a test document, we first extract from it (1) all single-word nouns and verbs and (2) all words and phrases that have appeared at least once as a trigger in the training data. We treat each of these extracted words and phrases as a candidate event mention.[3] The goal of the model is to make joint predictions for the candidate event mentions in a document. Three predictions will be made for each candidate event mention that correspond to the three tasks in the model: its trigger subtype, its anaphoricity, and its antecedent.

Given this formulation, we define three types of output variables:

- Event subtype variables $\mathbf{t} = (t_1, \ldots, t_n)$. Each $t_i$ takes a value in the set of 18 event subtypes defined in KBP 2016 or NONE, which indicates that the event mention is not a trigger.
- Anaphoricity variables $\mathbf{a} = (a_1, \ldots, a_n)$. Each $a_i$ is either ANAPHORIC or NOT ANAPHORIC.
- Coreference variables $\mathbf{c} = (c_1, \ldots, c_n)$, where $c_i \in \{1, \ldots, i-1, \text{NEW}\}$. In other words, the value of each $c_i$ is the id of its antecedent, which can be one of the preceding event mentions or NEW (if the event mention underlying $c_i$ starts a new cluster).

Each candidate event mention is associated with exactly one coreference variable, one event subtype variable, and one anaphoricity variable. Our model induces the following log-linear probability distribution over these variables:

$$p(\mathbf{t}, \mathbf{a}, \mathbf{c} | x; \Theta) \propto \exp(\sum_i \theta_i f_i(\mathbf{t}, \mathbf{a}, \mathbf{c}, x))$$

[3]According to the KBP annotation guidelines, each word may trigger multiple event mentions (e.g., *murder* can trigger two event mentions with subtypes Life.Die and Conflict.Attack). Hence, our treating each extracted word as a candidate event mention effectively prevents a word from triggering multiple event mentions. Rather than complicate model design by relaxing this simplifying assumption, we present an alternative, though partial, solution to this problem wherein we allow each event mention to be associated with multiple event subtypes. See the Appendix for details.

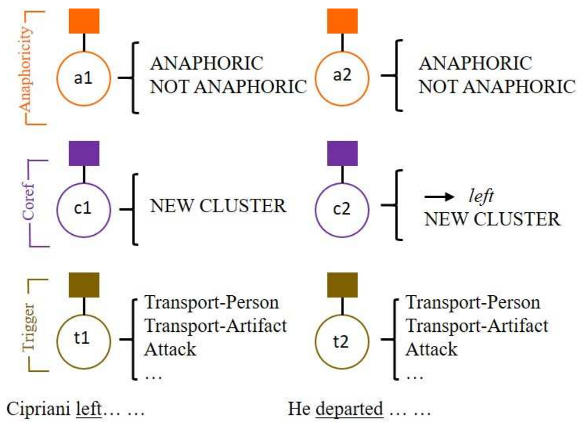

Figure 1: Unary factors for the three tasks, the variables they are connected to, and the possible values of the variables. Unary factors encode task-specific features. Each factor is connected to the corresponding output node. The features associated with a factor are used to predict the value of the output node it is connected to when a model is run independently of other models.

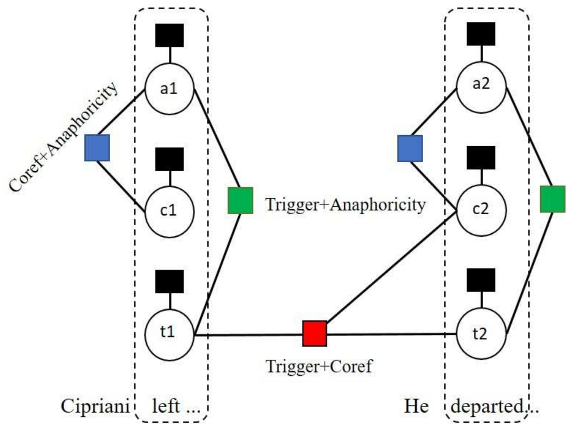

Figure 2: Binary and ternary factors. These higher-order factors capture cross-task interactions. The binary anaphoricity and trigger factors encourage anaphoric mentions to be triggers. The binary anaphoricity and coreference factors encourage non-anaphoric mentions to start a NEW coreference cluster. The ternary trigger and coreference factors encourage coreferent mentions to be triggers.

where $\theta_i \in \Theta$ is the weight associated with feature function $f_i$ and $x$ is the input document.

## 3.2 Features

Given that our model is a structured conditional random field, the features can be divided into two types: (1) task-specific features, and (2) cross-task features, which capture the interactions between a pair of tasks. We express these two types of features in factor graph notation. The task-specific features are encoded in unary factors, each of which is connected to the corresponding variable (Figure 1). The cross-task features are encoded in binary or ternary factors, each of which couples the output variables from two tasks (Figure 2). Next, we describe these two types of features. Each feature is used to train models for both English and Chinese unless otherwise stated.

### 3.2.1 Task-Specific Features

We begin by describing the task-specific features, which are encoded in unary factors, as well as each of the three independent models.

#### 3.2.1.1 Trigger Detection

When applied in isolation, our trigger detection model returns a distribution over possible subtypes given a candidate trigger. Each candidate trigger $t$ is represented using $t$'s word, $t$'s lemma, word bigrams formed with a window size of three from $t$, as well as feature conjunctions created by pairing $t$'s lemma with each of the following features: the head word of the entity syntactically closest to $t$, the head word of the entity textually closest to $t$, the entity type of the entity that is syntactically closest to $t$, and the entity type of the entity that is textually closest to $t$.[4] In addition, for event mentions with verb triggers, we use the head words and the entity types of their subjects and objects as features, where the subjects and objects are extracted from the dependency parse trees obtained using Stanford CoreNLP (Manning et al., 2014). For event mentions with noun triggers, we create the same features that we did for verb triggers, except that we replace the subjects and verbs with heuristically extracted agents and patients. Finally, for the Chinese trigger detector, we additionally create two features from each character in $t$, one encoding the character itself and the other encoding the entry number of the corresponding character in a Chinese synonym dictionary.[5]

#### 3.2.1.2 Event Coreference

We employ a mention-ranking model for event coreference that selects the most probable antecedent for a mention to be resolved (or NEW if the mention is non-anaphoric) from its set of candidate antecedents. When applied in isolation, the model is trained to maximize the condi-

[4]We train a CRF-based entity extraction model for jointly identifying the entity mentions and their types. Details can be found in Lu et al. (2016).

[5]The dictionary is available from http://ir.hit.edu.cn/. An entry number in this dictionary conceptually resembles a synset id in WordNet (Fellbaum, 1998).

tional likelihood of collectively resolving the mentions to their correct antecedents in the training texts (Durrett and Klein, 2013). Below we describe the features used to represent the candidate antecedents for the mention to be resolved, $m_j$.

**Features representing the NULL candidate antecedent:** Besides $m_j$'s word and $m_j$'s lemma, we employ feature conjunctions given their usefulness in entity coreference (Fernandes et al., 2014). Specifically, we create a conjunction between $m_j$'s lemma and the number of sentences preceding $m_j$, as well as a conjunction between $m_j$'s lemma and the number of mentions preceding $m_j$ in the document.

**Features representing a non-NULL candidate antecedent, $m_i$:** $m_i$'s word, $m_i$'s lemma, whether $m_i$ and $m_j$ have the same lemma, and feature conjunctions including: (1) $m_i$'s word paired with $m_j$'s word, (2) $m_i$'s lemma paired with $m_j$'s lemma, (3) the sentence distance between $m_i$ and $m_j$ paired with $m_i$'s lemma and $m_j$'s lemma, (4) the mention distance between $m_i$ and $m_j$ paired with $m_i$'s lemma and $m_j$'s lemma, (5) a quadruple consisting of $m_i$ and $m_j$'s subjects and their lemmas, and (6) a quadruple consisting of $m_i$ and $m_j$'s objects and their lemmas.

#### 3.2.1.3 Anaphoricity Determination

When used in isolation, the anaphoricity model returns the probability that the given event mention is anaphoric. To train the model, we represent each event mention $m_j$ using the following features: (1) the head word of each candidate antecedent paired with $m_j$'s word, (2) whether at least one candidate antecedent has the same lemma as that of $m_j$, and (3) the probability that $m_j$ is anaphoric in the training data (if $m_j$ never appears in the training data, this probability is set to 0.5).

### 3.2.2 Cross-Task Interaction Features

Cross-task interaction features are associated with the binary and ternary factors.

#### 3.2.2.1 Trigger Detection and Anaphoricity

We fire features that conjoin each candidate event mention's event subtype, the lemma of its trigger and its anaphoricity.

#### 3.2.2.2 Trigger Detection and Coreference

We define our joint coreference and trigger detection factors such that the features defined on subtype variables $t_i$ and $t_j$ are fired only if current mention $m_j$ is coreferent with preceding mention $m_i$. These features are: (1) the pair of $m_i$ and $m_j$'s subtypes, (2) the pair of $m_j$'s subtype and $m_i$'s word, and (3) the pair of $m_i$'s subtype and $m_j$'s word.

#### 3.2.2.3 Coreference and Anaphoricity

We fire a feature that conjoins event mention $m_j$'s anaphoricity and whether or not a non-NULL antecedent is selected for $m_j$.

## 3.3 Training

We learn the model parameters $\Theta$ from a set of $d$ training documents, where document $i$ contains content $x_i$, gold triggers $\mathbf{t_i^*}$ and gold event coreference partition $C_i^*$. Before learning, there are a couple of issues we need to address.

First, we need to derive gold anaphoricity labels $\mathbf{a_i^*}$ from $C_i^*$. This is straightforward: the first mention of each coreference chain is NOT ANAPHORIC, whereas the rest are ANAPHORIC.

Second, we employ gold event mentions for model training, but training models only on gold mentions is not sufficient: for instance, a trigger detector trained solely on gold mentions will not be able to classify a candidate event mention as NONE during testing. To address this issue, we additionally train the models on candidate event mentions corresponding to non-triggers. We create these candidate event mentions as follows. For each word $w$ that appears as a true trigger at least once in the training data, we create a candidate event mention from each occurrence of $w$ in the training data that is not annotated as a true trigger.

Third, since our model produces event coreference output in the form of an antecedent vector (with one antecedent per event mention), it needs to be trained on antecedent vectors. However, since the coreference annotation for each document $i$ is provided in the form of a clustering $C_i^*$, we follow previous work on entity coreference resolution (Durrett and Klein, 2013): we sum over all antecedent structures $A(C_i^*)$ that are *consistent* with $C_i^*$ (i.e., the first mention of a cluster has antecedent NEW, whereas each of the subsequent mentions can select any of the preceding mentions in the same cluster as its antecedent).

Next, we learn the model parameters to maximize the following conditional likelihood of the training data with L1 regularization:

$$L(\Theta) = \sum_{i=1}^{d} \log \sum_{\mathbf{c}^* \in A(C_i^*)} p'(\mathbf{t_i^*}, \mathbf{a_i^*}, \mathbf{c}^* | x_i; \Theta) + \lambda \|\Theta\|_1$$

In this objective, $p'$ is obtained by augmenting the distribution $p$ (defined in Section 3.1) with task-specific parameterized loss functions:

$$p'(\mathbf{t}, \mathbf{a}, \mathbf{c}|x_i; \Theta) \propto p(\mathbf{t}, \mathbf{a}, \mathbf{c}|x_i; \Theta) \exp[\alpha_t l_t(\mathbf{t}, \mathbf{t}^*) + \alpha_a l_a(\mathbf{a}, \mathbf{a}^*) + \alpha_c l_c(\mathbf{c}, C^*)]$$

where $l_t$, $l_a$ and $l_c$ are task-specific loss functions, and $\alpha_t$, $\alpha_a$ and $\alpha_c$ are the associated weight parameters that specify the relative importance of the three tasks in the objective function.

Softmax-margin, the technique of integrating task-specific loss functions into the objective function, was introduced by Gimpel and Smith (2010) and subsequently used by Durrett and Klein (2013, 2014). By encoding task-specific knowledge, these loss functions can help train a model that places less probability mass on less desirable output configurations.

Our loss function for event coreference, $l_c$, is motivated by the one Durrett and Klein (2013) developed for entity coreference. It is a weighted sum of the counts of three error types:

$$l_c(\mathbf{c}, C^*) = \alpha_{c,FA} FA(\mathbf{c}, C^*) + \alpha_{c,FN} FN(\mathbf{c}, C^*) + \alpha_{c,WL} WL(\mathbf{c}, C^*)$$

where $FA(\mathbf{c}, C^*)$ is the number of non-anaphoric mentions misclassified as anaphoric, $FN(\mathbf{c}, C^*)$ is the number of anaphoric mentions misclassified as non-anaphoric, and $WL(\mathbf{c}, C^*)$ is the number of incorrectly resolved anaphoric mentions.

Our loss function for trigger detection, $l_t$, is parameterized in a similar way, having three parameters associated with three error types: $\alpha_{t,FT}$ is associated with the number of non-triggers misclassified as triggers, $\alpha_{t,FN}$ is associated with the number of triggers misclassified as non-triggers, and $\alpha_{t,WL}$ is associated with the number of triggers labeled with the wrong subtype.

Finally, our loss function for anaphoricity determination, $l_a$, is also similarly parameterized, having two parameters: $\alpha_{a,FA}$ and $\alpha_{a,FN}$ are associated with the number of false anaphors and the number of false non-anaphors, respectively.

Following Durrett and Klein (2014), we use AdaGrad (Duchi et al., 2011) to optimize our objective with $\lambda = 0.001$ in our experiments.

### 3.4 Inference

Inference, which is performed during training and decoding, involves computing the marginals for a variable or a set of variables to which a factor connects. For efficiency, we perform approximate inference using belief propagation rather than exact inference. Given that convergence can typically be reached within five iterations of belief propagation, we employ five iterations in all experiments.

Performing inference using belief propagation on the full factor graph defined in Section 3.1 can still be computationally expensive, however. One reason is that the number of ternary factors grows quadratically with the number of event mentions in a document. To improve scalability, we restrict the domains of the coreference variables. Rather than allow the domain of coreference variable $c_j$ to be of size $j$, we allow a preceding mention $m_i$ to be a candidate antecedent of mention $m_j$ if (1) the sentence distance between the two mentions is less than an empirically determined threshold and (2) either they are coreferent at least once in the training data or their head words have the same lemma. Doing so effectively enables us to prune the unlikely candidate antecedents for each event mention. As Durrett and Klein (2014) point out, such pruning has the additional benefit of reducing "the memory footprint and time needed to build a factor graph", as we do not need to create any factor between $m_i$ and $m_j$ and its associated features if $m_i$ is pruned. To further reduce the memory footprint, we additionally restrict the domains of the event subtype variables. Given a candidate event mention created from word $w$, we allow the domain of its subtype variable to include only NONE as well as those subtypes that $w$ is labeled with at least once in the training data.

For decoding, we employ minimum Bayes risk, which computes the marginals of each variable w.r.t. the joint model and derives the most probable assignment to each variable.

## 4 Evaluation

### 4.1 Experimental Setup

We perform training and evaluation on the KBP 2016 English and Chinese corpora. For English, we train models on 509 of the training documents, tune parameters on 139 training documents, and report results on the official KBP 2016 English test set.[6] For Chinese, we train models on 302 of the training documents, tune parameters on 81 training documents, and report results on the official

[6]The parameters to be tuned are the $\alpha$'s multiplying the loss functions and those inside the loss functions.

|  | English |  |  |  |  |  |  |
|---|---|---|---|---|---|---|---|
|  | MUC | $B^3$ | CEAF$_e$ | BLANC | AVG-F | Trigger | Anaphoricity |
| KBP2016 | 26.37 | 37.49 | 34.21 | 22.25 | 30.08 | 46.99 | – |
| INDEP. | 22.71 | 40.72 | 39.00 | 22.71 | 31.28 | 48.82 | 27.35 |
| JOINT | 27.41 | 40.90 | 39.00 | 25.00 | 33.08 | 49.30 | 31.94 |
| $\Delta$ over INDEP. | +4.70 | +0.18 | 0.00 | +2.29 | +1.80 | +0.48 | +4.59 |
|  | Chinese |  |  |  |  |  |  |
|  | MUC | $B^3$ | CEAF$_e$ | BLANC | AVG-F | Trigger | Anaphoricity |
| KBP2016 | 24.27 | 32.83 | 30.82 | 17.80 | 26.43 | 40.01 | – |
| INDEP. | 22.68 | 32.97 | 29.96 | 17.74 | 25.84 | 39.82 | 19.31 |
| JOINT | 27.94 | 33.01 | 29.96 | 20.24 | 27.79 | 40.53 | 23.33 |
| $\Delta$ over INDEP. | +5.26 | +0.04 | 0.00 | +2.50 | +1.95 | +0.71 | +4.02 |

Table 2: Results of all three tasks on the KBP 2016 evaluation sets. The KBP2016 results are those achieved by the best-performing coreference resolver in the official KBP 2016 evaluation. $\Delta$ is the performance difference between the JOINT model and the corresponding INDEP. model. All results are expressed in terms of F-score.

KBP 2016 Chinese test set.

Results of event coreference and trigger detection are obtained using version 1.7.2 of the official scorer provided by the KBP 2016 organizers. To evaluate event coreference performance, the scorer employs four scoring measures, namely MUC (Vilain et al., 1995), $B^3$ (Bagga and Baldwin, 1998), CEAF$_e$ (Luo, 2005) and BLANC (Recasens and Hovy, 2011), as well as the unweighted average of their F-scores (AVG-F). The scorer reports event mention detection performance in terms of F-score, considering a mention correctly detected if it has an exact match with a gold mention in terms of boundary, event type, and event subtype. In addition, we report anaphoricity determination performance in terms of the F-score computed over anaphoric mentions, counting an extracted anaphoric mention as a true positive if it has an exact match with a gold anaphoric mention in terms of boundary.

### 4.2 Results and Discussion

Results are shown in Table 2 where performance on all three tasks (event coreference, trigger detection, and anaphoricity determination) is expressed in terms of F-score. The top half of the table shows the results on the English evaluation set. Specifically, row 1 shows the performance of the best event coreference system participating in KBP 2016 (Lu and Ng, 2016). This system adopts a pipeline architecture. It first uses an ensemble of one-nearest-neighbor classifiers for trigger detection. Using the extracted triggers, it then applies a pipeline of three sieves, each of which is a one-nearest-neighbor classifier, for event coreference. As we can see, this system achieves an AVG-F of 30.08 for event coreference and an F-score of 46.99 for trigger detection.

Row 2 shows the performance of the independent models, each of which is trained independently of the other models. Specifically, each independent model is trained using only the unary factors associated with it. As we can see, the independent models outperform the top KBP 2016 system by 1.2 points in AVG-F for event coreference and 1.83 points for trigger detection.

Results of our joint model are shown in row 3. The absolute performance differences between the joint model and the independent models are shown in row 4. As we can see, the joint model outperforms the independent models for all three tasks: by 1.80 points for event coreference, 0.48 points for trigger detection and 4.59 points for anaphoricity determination. Most encouragingly, the joint model outperforms the top KBP 2016 system for both event coreference and trigger detection. For event coreference, it outperforms the top KBP system w.r.t. all scoring metrics, yielding an improvement of 3 points in AVG-F. For trigger detection, it outperforms the top KBP system by 2.31 points.

The bottom half of Table 2 shows the results on the Chinese evaluation set. The top KBP 2016 event coreference system on Chinese is also the Lu and Ng (2016) system. While the top KBP system outperforms the independent models for both tasks (by 0.59 points in AVG-F for event coreference and 0.19 points for trigger detection), our joint model outperforms the independent models

| | English | | | Chinese | | |
|---|---|---|---|---|---|---|
| | Coref | Trigger | Anaph | Coref | Trigger | Anaph |
| INDEP. | 31.28 | 48.82 | 27.35 | 25.84 | 39.82 | 19.31 |
| INDEP.+CorefTrigger | +0.39 | +0.42 | −0.05 | +0.95 | +0.56 | −0.37 |
| INDEP.+CorefAnaph | +0.40 | −0.08 | +3.45 | +0.37 | +0.04 | −0.11 |
| INDEP.+TriggerAnaph | +0.11 | +0.38 | +1.35 | +0.14 | +0.52 | +0.02 |
| JOINT−CorefTrigger | +0.56 | −0.06 | +4.41 | +0.19 | +0.35 | +3.34 |
| JOINT−CorefAnaph | +0.63 | +0.66 | +1.46 | +1.50 | +0.88 | +0.28 |
| JOINT−TriggerAnaph | +1.89 | +0.50 | +4.01 | +1.65 | +0.50 | +1.79 |
| JOINT | +1.80 | +0.48 | +4.59 | +1.95 | +0.71 | +4.02 |

Table 3: Results of model ablations on the KBP 2016 evaluation sets. Each row of ablation results is obtained by either adding one type of interaction factor to the INDEP. model or deleting one type of interaction factor from the JOINT model. For each column, the results are expressed in terms of changes to the INDEP. model's F-score shown in row 1.

for all three tasks: by 1.95 points for event coreference, 4.02 points for anaphoricity determination, and 0.71 points for trigger detection. Like its English counterpart, our Chinese joint model outperforms the top KBP system for both event coreference and trigger detection. For event coreference, it outperforms the top KBP system w.r.t. all but the CEAF$_e$ metric, yielding an absolute improvement of 1.36 points in AVG-F. For trigger detection, it outperforms the top KBP system by 0.52 points.

For both datasets, the joint model's superior performance to the independent coreference model stems primarily from considerable improvements in MUC F-score. As MUC is a link-based measure, these results provide suggestive evidence that joint modeling has enabled more event coreference links to be discovered.

## 4.3 Model Ablations

To evaluate the importance of each of the three types of joint factors in the joint model, we perform ablation experiments.[7] Table 3 shows the results on the English and Chinese datasets when we add each type of joint factors to the independent model and remove each type of joint factors from the full joint model. The results of each task are expressed in terms of changes to the corresponding independent model's F-score.

**Coref-Trigger interactions.** Among the three types of factors, this one contributes the most to coreference performance, regardless of whether it is applied in isolation or in combination with the other two types of factors to the independent coreference model. In addition, it is the most effective type of factor for improving trigger detection. When applied in combination, it also improves anaphoricity determination, although less effectively than the other two types of factors.

**Coref-Anaphoricity interactions.** When applied in isolation to the independent models, this type of factor improves coreference resolution but has a mixed impact on anaphoricity determination. When applied in combination with other types of factors, it improves both tasks, particularly anaphoricity determination. Its impact on trigger detection, however, is generally negative.

**Trigger-Anaphoricity interactions.** When applied in isolation to the independent models, this type of factor improves both trigger detection and anaphoricity determination. When applied in combination with other types of factors, it still improves anaphoricity determination (particularly on Chinese), but has a mixed effect on trigger detection. Among the three types of factors, it has the least impact on coreference resolution.

## 4.4 Error Analysis

Next, we conduct an analysis of the major sources of error made by our joint coreference model.

### 4.4.1 Two Major Types of Precision Error

**Erroneous and mistyped triggers.** Our trigger model tends to assign the same subtype to event mentions triggered by the same word. As a result, it often assigns the wrong subtype to triggers that

[7]Chen and Ng (2013) also performed ablation on their ACE-style Chinese event coreference resolver. However, given the differences in the tasks involved (e.g., they did not model event anaphoricity, but included tasks such as event argument extraction and role classification, entity coreference, and event mention attribute value computation) and the ablation setup (e.g., they ablated individual tasks/components in their pipeline-based system in an incremental fashion, whereas we ablate interaction factors rather than tasks), a direct comparison of their observations and ours is difficult.

possess different subtypes in different contexts. For the same reason, words that are only sometimes used as triggers are often wrongly posited as triggers when they are not. These two types of triggers have in turn led to the establishment of incorrect coreference links.[8]

**Failure to extract arguments.** In the absence of an annotated corpus for training an argument classifier, we exploit dependency relations for argument extraction. Doing so proves inadequate, particularly for noun triggers, owing to the absence of dependency relations that can be used to reliably extract their arguments. Moreover, using dependency relations does not allow the extraction of arguments that do not appear in the same sentence as their trigger. Since the presence of incompatible arguments is an important indicator of non-coreference, our model's failure to extract arguments has resulted in incorrect coreference links.

#### 4.4.2 Three Major Types of Recall Error

**Missing triggers.** Our trigger model fails to identify trigger words that are unseen or rarely-occurring in the training data. As a result, many coreference links cannot be established.

**Lack of entity coreference information.** Entity coreference information is useful for event coreference because the corresponding arguments of two event mentions are typically coreferent. Since our model does not exploit entity coreference information, it treats two lexically different event arguments as non-coreferent/unrelated. This in turn weakens its ability to determine whether two event mentions are coreferent. This issue is particularly serious in discussion forum documents, where it is not uncommon to see pronouns serve as subjects and objects of event mentions. The situation is further aggravated in Chinese documents, where zero pronouns are prevalent.

**Lack of contextual understanding.** Our model only extracts features from the sentence in which an event mention appears. However, additional contextual information present in neighboring sentences may be needed for correct coreference resolution. This is particularly true in discussion forum documents, where the same event may be described differently by different people. For example, when describing the fact that Tim Cook will attend Apple's Istanbul store opening, one person said "Cook is expected to return to Turkey for the store opening", and another person described this event as "Tim travels abroad YET AGAIN to be feted by the not-so-high-and-mighty". It is by no means easy to determine that *return* and *travel* trigger two coreferent mentions in these sentences.

[8]In our joint model, mentions that are posited as coreferent are encouraged to have the same subtype. While it can potentially fix the errors involving coreferent mentions that have different subtypes, it cannot fix the errors in which the two mentions involved have the same erroneous subtype.

## 5 Related Work

Existing event coreference resolvers have been evaluated on different corpora, such as MUC (e.g., Humphreys et al. (1997)), ACE (e.g., Ahn (2006), Chen and Ji (2009), McConky et al. (2012), Sangeetha and Arock (2012), Chen and Ng (2015, 2016), Krause et al. (2016)), OntoNotes (e.g., Chen et al. (2011)), the Intelligence Community corpus (e.g., Cybulska and Vossen (2012), Araki et al. (2014), Liu et al. (2014)), the ECB corpus (e.g., Lee et al. (2012), Bejan and Harabagiu (2014)) and its extension ECB+ (e.g., Yang et al. (2015)), and ProcessBank (e.g., Araki and Mitamura (2015)). The newest event coreference corpora are the ones used in the KBP 2015 and 2016 Event Nugget Detection and Coreference shared tasks, in which the best performers in 2015 and 2016 are RPI's system (Hong et al., 2015) and UTD's system (Lu and Ng, 2016), respectively. The KBP 2015 corpus has recently been used to evaluate Peng et al.'s (2016) minimally supervised approach and Lu et al.'s (2016) joint inference approach to event coreference. With the rarest exceptions (e.g., Lu et al. (2016)), existing resolvers have adopted a pipeline architecture in which trigger detection is performed prior to coreference resolution.

## 6 Conclusion

We proposed a joint model of event coreference resolution, trigger detection, and event anaphoricity determination. The model is novel in its choice of tasks and the cross-task interaction features. When evaluated on the KBP 2016 English and Chinese corpora, our model not only outperforms the independent models but also achieves the best results to date on these corpora.

## Acknowledgments

We thank the three anonymous reviewers for their detailed comments. This work was supported in part by NSF Grants IIS-1219142 and IIS-1528037.

## References

David Ahn. 2006. The stages of event extraction. In *Proceedings of the COLING/ACL Workshop on Annotating and Reasoning about Time and Events*. pages 1–8.

Jun Araki, Zhengzhong Liu, Eduard Hovy, and Teruko Mitamura. 2014. Detecting subevent structure for event coreference resolution. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation*, pages 4553–4558.

Jun Araki and Teruko Mitamura. 2015. Joint event trigger identification and event coreference resolution with structured perceptron. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2074–2080.

Amit Bagga and Breck Baldwin. 1998. Algorithms for scoring coreference chains. In *Proceedings of the Linguistic Coreference Workshop at The First International Conference on Language Resources and Evaluation*, pages 563–566.

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

Bin Chen, Jian Su, Sinno Jialin Pan, and Chew Lim Tan. 2011. A unified event coreference resolution by integrating multiple resolvers. In *Proceedings of the Fifth International Conference on Natural Language Processing*. pages 102–110.

Chen Chen and Vincent Ng. 2013. Chinese event coreference resolution: Understanding the state of the art. In *Proceedings of the 6th International Joint Conference on Natural Language Processing*. pages 822–828.

Chen Chen and Vincent Ng. 2015. Chinese event coreference resolution: An unsupervised probabilistic model rivaling supervised resolvers. In *Proceedings of Human Language Technologies: The 2015 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. pages 1097–1107.

Chen Chen and Vincent Ng. 2016. Joint inference over a lightly supervised information extraction pipeline: Towards event coreference resolution for resource-scarce languages. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence*. pages 2913–2920.

Zheng Chen and Heng Ji. 2009. Graph-based event coreference resolution. In *Proceedings of the 2009 Workshop on Graph-based Methods for Natural Language Processing (TextGraphs-4)*, pages 54–57.

Agata Cybulska and Piek Vossen. 2012. Using semantic relations to solve event coreference in text. In *Proceedings of the LREC Workshop on Semantic Relations-II Enhancing Resources and Applications*, pages 60–67.

John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research* 12:2121–2159.

Greg Durrett and Dan Klein. 2013. Easy victories and uphill battles in coreference resolution. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1971–1982.

Greg Durrett and Dan Klein. 2014. A joint model for entity analysis: Coreference, typing, and linking. *Transactions of the Association for Computational Linguistics* 2:477–490.

Christiane Fellbaum. 1998. *WordNet: An Electronical Lexical Database*. MIT Press, Cambridge, MA.

Eraldo Rezende Fernandes, Cícero Nogueira dos Santos, and Ruy Luiz Milidiu. 2014. Latent trees for coreference resolution. *Computational Linguistics* 40(4):801–835.

Kevin Gimpel and Noah A Smith. 2010. Softmax-margin CRFs: Training log-linear models with cost functions. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 733–736.

Yu Hong, Di Lu, Dian Yu, Xiaoman Pan, Xiaobin Wang, Yadong Chen, Lifu Huang, and Heng Ji. 2015. RPI BLENDER TAC-KBP2015 system description. In *Proceedings of the Eighth Text Analysis Conference*.

Kevin Humphreys, Robert Gaizauskas, and Saliha Azzam. 1997. Event coreference for information extraction. In *Proceedings of the ACL/EACL Workshop on Operational Factors in Practical, Robust Anaphora Resolution for Unrestricted Texts*, pages 75–81.

Sebastian Krause, Feiyu Xu, Hans Uszkoreit, and Dirk Weissenborn. 2016. Event linking with sentential features from convolutional neural networks. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning*, pages 239–249.

Heeyoung Lee, Marta Recasens, Angel Chang, Mihai Surdeanu, and Dan Jurafsky. 2012. Joint entity and event coreference resolution across documents. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 489–500.

Zhengzhong Liu, Jun Araki, Eduard Hovy, and Teruko Mitamura. 2014. Supervised within-document event coreference using information propagation. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation*, pages 4539–4544.

Zhengzhong Liu, Jun Araki, Teruko Mitamura, and Eduard Hovy. 2016. CMU-LTI at KBP 2016 event nugget track. In *Proceedings of the Ninth Text Analysis Conference*.

Jing Lu and Vincent Ng. 2016. UTD's event nugget detection and coreference system at KBP 2016. In *Proceedings of the Ninth Text Analysis Conference*.

Jing Lu, Deepak Venugopal, Vibhav Gogate, and Vincent Ng. 2016. Joint inference for event coreference resolution. In *Proceedings of the 26th International Conference on Computational Linguistics*, pages 3264–3275.

Xiaoqiang Luo. 2005. On coreference resolution performance metrics. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 25–32.

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60.

Katie McConky, Rakesh Nagi, Moises Sudit, and William Hughes. 2012. Improving event coreference by context extraction and dynamic feature weighting. In *Proceedings of the 2012 IEEE International Multi-Disciplinary Conference on Cognitive Methods in Situation Awareness and Decision Support*, pages 38–43.

Teruko Mitamura, Zhengzhong Liu, and Eduard Hovy. 2016. Overview of TAC-KBP 2016 event nugget track. In *Proceedings of the Ninth Text Analysis Conference*.

Vincent Ng. 2010. Supervised noun phrase coreference research: The first fifteen years. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*. pages 1396–1411.

Thien Huu Nguyen, Adam Meyers, and Ralph Grishman. 2016. New York University 2016 system for KBP event nugget: A deep learning approach. In *Proceedings of Ninth Text Analysis Conference*.

Haoruo Peng, Yangqiu Song, and Dan Roth. 2016. Event detection and co-reference with minimal supervision. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. pages 392–402.

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

S. Sangeetha and Michael Arock. 2012. Event coreference resolution using mincut based graph clustering. In *Proceedings of the Fourth International Workshop on Computer Networks & Communications* pages 253–260.

Zhiyi Song, Ann Bies, Stephanie Strassel, Tom Riese, Justin Mott, Joe Ellis, Jonathan Wright, Seth Kulick, Neville Ryant, and Xiaoyi Ma. 2015. From light to rich ERE: Annotation of entities, relations, and events. In *Proceedings of the 3rd Workshop on EVENTS*, pages 89–98.

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

Sam Wiseman, Alexander M. Rush, Stuart Shieber, and Jason Weston. 2015. Learning anaphoricity and antecedent ranking features for coreference resolution. 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 1416–1426.

Bishan Yang, Claire Cardie, and Peter Frazier. 2015. A hierarchical distance-dependent Bayesian model for event coreference resolution. *Transactions of the Association for Computational Linguistics* 3:517–528.

## Appendix: Handling Words that Trigger Multiple Event Mentions

In KBP, a word can trigger multiple event mentions. However, since we create exactly one candidate event mention from each extracted word in each test document, our model effectively prevents a word from triggering multiple event mentions. This poses a problem: each word cannot be associated with more than one event subtype. This appendix describes how we (partially) address this issue that involves allowing each event mention to be associated with multiple event subtypes.

To address this problem, we preprocess the *gold trigger annotations* in the *training* data as follows. First, for each word triggering multiple event mentions (with different event subtypes), we merge their event mentions into one event mention having the combined subtype. In principle, we can add each of these combined subtypes into our event subtype inventory and allow our model to make predictions using them. However, to avoid over-complicating the prediction task (by having a large subtype inventory), we only add the three most frequently occurring combined subtypes in the training data to the inventory. Merged mentions whose combined subtype is not among the most frequent three will be unmerged in order to recover the original mentions so that the model can still be trained on them.

To train our joint model, however, the trigger annotations and the event coreference annotations in the training data must be consistent. Since we modified the trigger annotations (by merging event mentions and allowing combined subtypes), we make two modifications to the event coreference annotations to ensure consistency between the two sets of annotations. First, let $C_1$ and $C_2$ be two event coreference chains in a training document such that the set of words triggering the event mentions in $C_1$ (with subtype $t_1$) is the same as that triggering the event mentions in $C_2$ (with subtype $t_2$). If each of the event mentions in $C_1$ was merged with the corresponding event mention in $C_2$ during the aforementioned preprocessing of the trigger annotations (because combining $t_1$ and $t_2$ results in one of the three most frequent combined subtypes), then we delete one of the two coreference chains, and assign the combined subtype to the remaining chain. Finally, we remove any remaining event mentions that were merged during the preprocessing of trigger annotations from their respective coreference chains and create a singleton cluster for each of the merged mentions.