Improving Event Coreference Resolution by Learning Argument Compatibility from Unlabeled Data

Yin Jou Huang[1], **Jing Lu**[2], **Sadao Kurohashi**[1], **Vincent Ng**[2]
[1]Graduate School of Informatics, Kyoto University
[2]Human Language Technology Research Institute, University of Texas at Dallas
huang@nlp.ist.i.kyoto-u.ac.jp, kuro@i.kyoto-u.ac.jp
{ljwinnie,vince}@hlt.utdallas.edu

Abstract

Argument compatibility is a linguistic condition that is frequently incorporated into modern event coreference resolution systems. If two event mentions have incompatible arguments in any of the argument roles, they cannot be coreferent. On the other hand, if these mentions have compatible arguments, then this may be used as information toward deciding their coreferent status. One of the key challenges in leveraging argument compatibility lies in the paucity of labeled data. In this work, we propose a transfer learning framework for event coreference resolution that utilizes a large amount of unlabeled data to learn the argument compatibility between two event mentions. In addition, we adopt an interactive inference network based model to better capture the (in)compatible relations between the context words of two event mentions. Our experiments on the KBP 2017 English dataset confirm the effectiveness of our model in learning argument compatibility, which in turn improves the performance of the overall event coreference model.

1 Introduction

Events are essential building blocks of all kinds of natural language text. An event can be described several times from different aspects in the same document, resulting in multiple surface forms of event mentions. The goal of event coreference resolution is to identify event mentions that correspond to the same real-world event. This task is critical for natural language processing applications that require deep text understanding, such as storyline extraction/generation, text summarization, question answering, and information extraction.

Figure 1 shows a document consisting of three events described by six different event mentions. Among these event mentions, m_1, m_2 and m_4 are coreferent, since they all correspond to the event of the KMT party electing a new party chief. Similarly, m_3 and m_5 are also coreferent, while m_6 is not coreferent with any other event mentions.

[1]	KMT to **elect**$_{m1}$ new party chief
[2]	The ruling Chinese Kuomingtang Party (KMT) in Taiwan is scheduled to **elect**$_{m2}$ a new chairperson in January 2012, it was decided at the top-level KMT **meeting**$_{m3}$ on Wednesday.
[3]	KMT chairman Ma Ying-jeou stressed fair paly during the **election**$_{m4}$ at Wednesday's **meeting**$_{m5}$, saying guest-feting or gift-giving must be banned during the process.
[4]	Ma was twice **elected**$_{m6}$ the party's chairman, first in 2005 and then in 2009.
ev_1: m_1, m_2, m_4 ev_2: m_3, m_5 ev_3: m_6	

Figure 1: A document with three events described in six event mentions. Coreferent event mentions are highlighted with the same color.

An event mention consists of a trigger and zero or more arguments. The trigger of an event mention is the word/phrase that is considered the most representative of the event, such as the word *meeting* for m_3 or the word *elected* for m_6. Triggers of coreferent event mentions must be related, that is, they should describe the same type of events. For example, m_1 and m_3 cannot be coreferent, since their trigger words — $elect$ and $meeting$ — are not related.

Arguments are the participants of an event, each having its role. For example, *KMT* is the AGENT-argument and *new party chief* is the PATIENT-argument of m_1. Argument compatibility is an important linguistic condition for determining the coreferent status between two event mentions. Two arguments are incompatible if they do not correspond to the same real-world entity when they are expressed in the same level of specificity;

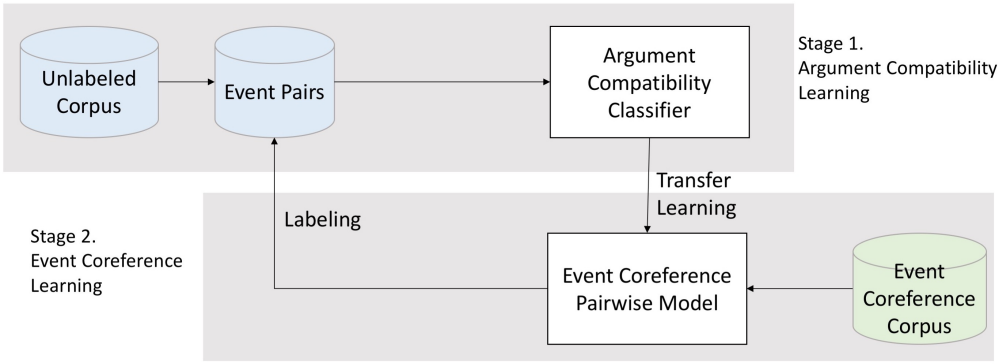

Figure 2: System overview.

otherwise, they are compatible. For example, a pair of TIME-arguments — *Wednesday* and *2005* — which are expressed in different level of specificity, are considered compatible. If two event mentions have incompatible arguments in some specific argument roles, they cannot be coreferent. For example, m_2 and m_6 are not coreferent since their TIME-arguments — *January 2012* and *2005* — and their PATIENT-arguments — *a new chairperson* and *Ma* — are incompatible. On the other hand, coreferent event mentions can only have compatible arguments. For example, m_3 and m_5 both have *Wednesday* as TIME-arguments. In this example, argument compatibility in the TIME argument role is a strong hint suggesting their coreference.

Despite its importance, incorporating argument compatibility into event coreference systems is challenging due to the lack of sufficient labeled data. Many existing works have relied on implementing argument extractors as upstream components and designing argument features that capture argument compatibility in event coreference resolvers. However, the error introduced in each of the steps propagates through these resolvers and hinders their performance considerably.

In light of the aforementioned challenge, we propose a framework for transferring argument (in)compatibility knowledge to the event coreference resolution system, specifically by adopting the interactive inference network (Gong et al., 2018) as our model structure. The idea is as follows. First, we train a network to determine whether the corresponding arguments of an event mention pair are compatible on automatically labeled training instances collected from a large unlabeled news corpus. Second, to transfer the knowledge of argument (in)compatibility to an event coreference resolver, we employ the network (pre)trained in the previous step as a starting point and train it to determine whether two event mentions are coreferent on manually labeled event coreference corpora. Third, we iteratively repeat the above two steps, where we use the learned coreference model to relabel the argument compatibility instances, retrain the network to determine argument compatibility, and use the resulting pretrained network to learn an event coreference resolver. In essence, we mutually bootstrap the argument (in)compatibility determination task and the event coreference resolution task.

Our contributions are three-fold. First, we utilize and leverage the argument (in)compatibility knowledge acquired from a large unlabeled corpus for event coreference resolution. Second, we employ the interactive inference network as our model structure to iteratively learn argument compatibility and event coreference resolution. Initially proposed for the task of natural language inference, the interactive inference network is suitable for capturing the semantic relations between word pairs. Experimental results on the KBP coreference dataset show that this network architecture is also suitable for capturing the argument compatibility between event mentions. Third, our model achieves state-of-the-art results on the KBP 2017 English dataset (Ellis et al., 2015, 2016; Getman et al., 2017), which confirms the effectiveness of our method.

2 Related Work

Ablation experiments conducted by Chen and Ng (2013) provide empirical support for the usefulness of event arguments for event coreference resolution. Hence, it should not be surprising that, with just a few exceptions (e.g., Sangeetha and

	event mention	DATE-compatibility with m_a
m_a	The result of the **election** last October surprised everyone.	-
m_1	He was **elected** as president in 2005.	no
m_2	The presidential **election** took place on October 20th.	yes
m_3	The opposition party won the **election**.	yes

Table 1: Examples of NER-based sample filtering. The phrases tagged as DATE are underlined, and the trigger words are boldfaced.

Arock (2012); Araki and Mitamura (2015); Lu and Ng (2017)), argument features have been extensively exploited in event coreference systems to capture the argument compatibility between two event mentions. Basic features such as the number of overlapping arguments and the number of unique arguments, and a binary feature encoding whether arguments are conflicting have been proposed (Chen et al., 2009; Chen and Ji, 2009; Chen and Ng, 2016). More sophisticated features based on different kinds of similarity measures have also been considered, such as the surface similarity based on Dice coefficient and the WuPalmer WordNet similarity between argument heads (McConky et al., 2012; Cybulska and Vossen, 2013; Araki et al., 2014; Liu et al., 2014; Krause et al., 2016). However, these features are computed using either the outputs of event argument extractors and entity coreference resolvers (Ahn, 2006; Chen and Ng, 2014, 2015; Lu and Ng, 2016) or semantic parsers (Bejan and Harabagiu, 2014; Yang et al., 2015; Peng et al., 2016) and therefore suffer from serious error propagation issues (see Lu and Ng (2018)). Several previous works proposed joint models to address this problem (Lee et al., 2012; Lu et al., 2016), while others utilized iterative methods to propagate argument information (Liu et al., 2014; Choubey and Huang, 2017) in order to alleviate this issue. However, all of these methods still rely on argument extractors to identify arguments and their roles.

3 Method

Our proposed transfer learning framework consists of two learning stages, the pretraining stage of an argument compatibility classifier and the fine-tuning stage of an event coreference resolver (Figure 2). We provide the details of both stages in sections 3.1 and 3.2, and describe the iterative strategy combining the two training stages in section 3.3. Details on the model structure are covered in section 3.4.

3.1 Argument Compatibility Learning

In the pretraining stage, we train the model as an argument compatibility classifier with event mentions extracted from a large unlabeled news corpus.

Task definition Given a pair of event mentions (m_a, m_b) with related triggers, predict whether their arguments are compatible or not.

Here, an event mention is represented by a trigger word and the context words within an n-word window around the trigger.

Related trigger extraction We analyze the event coreference resolution corpus and extract trigger pairs that are coreferent more than k times in the training data. We define these trigger pairs to be *related* triggers in our experiment. In this work, we set k to 10. Table 2 shows some examples of related triggers with high counts.

trigger pair	count
kill - death	86
shoot - shooting	35
retire - retire	34
demonstration - protest	30

Table 2: Examples of related triggers.

If the triggers of an event mention pair are related, their coreferent status cannot be determined by looking at the triggers alone, and this is the case in which argument compatibility affects the coreferent status most directly. Thus, we focus on the event mention pairs with related triggers in the pretraining stage of argument compatibility learning.

Compatible samples extraction From each document, we extract event mention pairs with related triggers and check whether the following conditions are satisfied:

1. DATE-compatibility (Table 1):
   First, we perform named entity recognition

(NER) on the context words. If both event mentions have phrases tagged as DATE in the context, these two phrases must contain at least one overlapping word. If there are multiple phrases tagged as DATE in the context, only the phrase closest to the trigger word is considered.

2. PERSON-compatibility: Similar to 1.
3. NUMBER-compatibility: Similar to 1.
4. LOCATION-compatibility: Similar to 1.
5. Apart from function words, the ratio of overlapping words in their contexts must be under 0.3 for both event mentions. We add this constraint in order to remove trivial samples of nearly identical sentences.

Conditions 1–4 are heuristic filtering rules based on NER tags, which aim to remove samples with apparent incompatibilities. Here, we consider four NER types — DATE, PERSON, NUMBER, and LOCATION — because these types of words are the most salient types of incompatibility that can be observed between event mentions. Condition 5 aims to remove event mention pairs that are "too similar". We add this condition because we do not want our model to base its decisions on the number of overlapping words between the event mentions.

We collect event mention pairs satisfying all the above conditions as our initial set of compatible samples.

Incompatible sample extraction From different documents in the corpus, we extract event mentions with related triggers and check whether the following conditions are satisfied:

1. The creation date of the two documents must be at least one month apart.
2. Apart from the trigger words and the function words, the context of the event mentions must contain at least one overlapping word.

In the unlabeled news corpus, articles describing similar news events are sometimes present. Thus, we use condition 1 to roughly assure that the event mention pairs extracted are not coreferent. Mention pairs extracted from the same document tend to contain overlapping content words, so to prevent our model to make decisions based on the existence of overlapping words, we add condition 2 as a constraint.

We collect event mention pairs satisfying all the above conditions as our initial set of incompatible samples.

Argument compatibility classifier With the initial set of compatible and incompatible samples acquired above, we train a binary classier to distinguish between samples of the two sets.

3.2 Event Coreference Learning

In the fine-tuning stage, we adapt the argument compatibility classifier on the labeled event coreference data to a mention-pair event coreference model.

3.2.1 Event Mention Detection

Before proceeding to the task of event coreference resolution, we have to identify the event mentions in the documents. We train a separate event mention detection model to identify event mentions along with their subtypes.

We model event mention detection as a multi-class classification problem. Given a candidate word along with its context, we predict the subtype of the event mention triggered by the word. If the given candidate word is not a trigger, we label it as NULL. We select the words that have appeared as a trigger at least once in the training data as candidate trigger words. We do not consider multi-word triggers in this work.

Given an input sentence, we first represent each of its comprising words by the concatenation of the word embedding and the character embedding of the word. These representation vectors are fed into a bidirectional LSTM (biLSTM) layer to obtain the hidden representation of each word.

For each candidate word in the sentence, its hidden representation is fed into the inference layer to predict the class label. Since the class distribution is highly unbalanced, with the NULL label significantly outnumbering all the other labels, we use a weighted softmax at the inference layer to obtain the probability of each class. In this work, we set the weight to 0.1 for the NULL class label and 1 for all the other class labels.

Intuitively, candidate triggers with the same surface form in the same document tend to have the same class label. However, it is difficult to model this consistency since our model operates at the sentence level. Thus, we account for this con-

sistency across sentences by the following post-processing step: If a candidate word is assigned the NULL label but more than half of the candidates sharing the same surface form is detected as triggers of a specific subtype, then we change the label to this given subtype. Also, we disregard event mentions with types *contact*, *movement* and *transaction* in this post-processing step, since the subtypes under these three types do not have a good consistency across different sentences in the same document.

3.2.2 Mention-Pair Event Coreference Model

With the argument compatibility classifier trained in the previous stage, we use the labeled event coreference corpus to fine-tune the model into an event coreference resolver. We design the event coreference resolver to be a mention-pair model (Soon et al., 2001), which takes a pair of event mentions as the input and outputs the likelihood of them being coreferent.

With the pairwise event coreference predictions, we further conduct best-first clustering (Ng and Cardie, 2002) on the pairwise results to build the event coreference clusters of each document. Best-first clustering is an agglomerative clustering algorithm that links each event mention to the antecedent event mention with the highest coreference likelihood given the likelihood is above an empirically determined threshold.

3.3 Iterative Relabeling Strategy

Previously, we collected a set of compatible event mentions from the same document with simple heuristic filtering. Despite this filtering step, the initial compatible set is noisy. Here, we introduce an iterative relabeling strategy to improve the quality of the compatible set of event mentions.

First, we calculate the coreference likelihood of the event mentions in the initial compatible set. Mention pairs with a coreference likelihood above threshold θ_M are added to the new compatible set. On the other hand, mention pairs with a coreference likelihood below θ_m are added to the initial incompatible set to form the new incompatible set. With the new compatible and incompatible sets, we can start another iteration of transfer learning to train a coreference resolver with improved quality. In this work, we set θ_M to 0.8 and θ_m to 0.2.

3.4 Model Structure

We adopt an interactive inference network as the model structure of our proposed method (Figure 3). A qualitative analysis of an interactive inference network shows that it is good at capturing word overlaps, antonyms and paraphrases between sentence pairs (Gong et al., 2018). Thus, we believe this network is suitable for capturing the argument compatibility between two event mentions. The model consists of the following components:

Model inputs The input to the model is a pair of event mentions (m_a, m_b), with m_a being the antecedent mention of m_b:

$$\begin{aligned} m_a &= \{w_a^1, w_a^2, ..., w_a^N\} \\ m_b &= \{w_b^1, w_b^2, ..., w_b^N\} \end{aligned} \quad (1)$$

Each event mention is represented by a sequence of N tokens consisting of one trigger word and its context. Here, we take the context to be the words within an n-word window around the trigger. In this work, n is set to 10.

Embedding layer We represent each input token by the concatenation of the following components:

Word embedding The word representation of the given token. We use pretrained word vectors to initialize the word embedding layer.

Character embedding To identify (in)compatibilities regarding person, organization or location names, the handling of out-of-vocabulary (OOV) words is critical.

Adding character-level embeddings can alleviate the OOV problem (Yang et al., 2017). Thus, we apply a convolutional neural network over the comprising characters of each token to acquire the corresponding character embedding.

POS and NER one-hot vectors One-hot vectors of the part-of-speech (POS) tag and NER tag.

Exact match A binary feature indicating whether a given token appears in the context of both event mentions. This feature is proved useful for several NLP tasks operating on pairs of texts (Chen et al., 2017; Gong et al., 2018; Pan et al., 2018).

Trigger position We encode the position of the trigger word by adding a binary feature to indicate whether a given token is a trigger word.

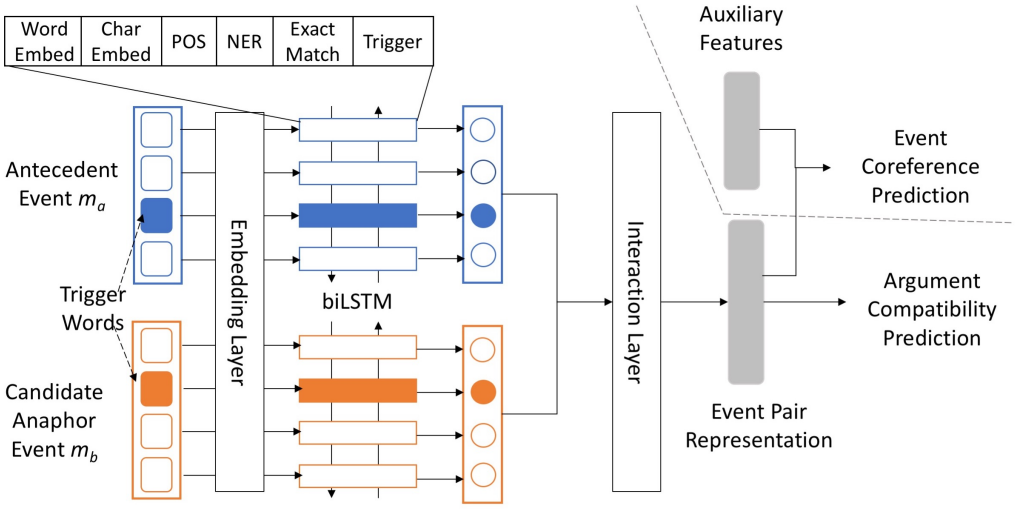

Figure 3: Model structure.

Encoding layer We pass the sequence of embedding vectors into a biLSTM layer (Hochreiter and Schmidhuber, 1997), resulting in a sequence of hidden vectors of size $|h|$:

$$h_a^i = biLSTM(emb(w_a^i), h_a^{i-1})$$
$$h_b^i = biLSTM(emb(w_b^i), h_b^{i-1}) \quad (2)$$

where $emb(w)$ is the embedding vector of token w.

Interaction layer The interaction layer captures the relations between two event mentions based on the hidden vectors h_a and h_b. The interaction tensor I, a 3-D tensor of shape (N, N, $|h|$), is calculated by taking the pairwise multiplication of the corresponding hidden vectors:

$$I_{ij} = h_a^i \circ h_b^j \quad (3)$$

Finally, we apply a multi-layer convolutional neural network to extract the event pair representation vector f_{ev}.

Inference layer In the pretraining stage, we feed f_{ev} to a fully-connected inference layer to make a binary prediction of argument compatibility.

As for the fine-tuning stage, we concatenate an auxiliary feature vector f_{aux} to f_{ev} before feeding it into the inference layer. f_{aux} consists of two features, a one-hot vector that encodes the sentence distance between the two event mentions and the difference of the word embedding vectors of the two triggers.

4 Evaluation

4.1 Experimental Setup

4.1.1 Corpora

We use English Gigaword (Parker et al., 2009) as the unlabeled corpus for argument compatibility learning. This corpus consists of the news articles from five news sources, each annotated with its creation date.

As for event coreference resolution, we use the English portion of the KBP 2015 and 2016 datasets (Ellis et al., 2015, 2016) for training, and the KBP 2017 dataset (Getman et al., 2017) for evaluation. The KBP datasets comprise news articles and discussion forum threads. The KBP 2015, 2016, and 2017 corpora contain 648, 169, and 167 documents, respectively. Each document is annotated with event mentions of 9 types and 18 subtypes, along with the coreference clusters of these event mentions.

4.1.2 Implementation Details

Preprocessing We use the Stanford CoreNLP toolkit (Manning et al., 2014) to perform preprocessing on the input data.

Network structure Each word embedding is initialized with the 300-dimensional pretrained GloVe embedding (Pennington et al., 2014). The character embedding layer is a combination of an 8-dimensional embedding layer and three 1D convolution layers with a kernel size of 5 with 100 filters. The size of the biLSTM layer is 200. The maximum length of a word is 16 characters; shorter words are padded with zero and longer

	MUC	B^3	$CEAF_e$	BLANC	AVG-F
biLSTM (standard)	29.49	43.15	39.91	24.15	34.18
biLSTM (transfer)	33.84	42.91	38.39	26.59	35.43
Interact (standard)	31.12	42.84	39.01	24.99	34.49
Interact (transfer)	34.28	42.93	39.95	32.12	36.24
Interact (transfer, 2^{nd} iter)	35.66	43.20	40.02	32.43	**36.75**
Interact (transfer, 3^{rd} iter)	36.05	43.07	39.69	28.06	36.72
Jiang et al. (2017)	30.63	43.84	39.86	26.97	35.33

Table 3: Event coreference resolution results of our proposed system, compared with the biLSTM baseline model and the current state-of-the-art system.

words are cropped. For the interaction layer, we use convolution layers with a kernel size of 3 in combination with max-pooling layers. The size of the inference layer is 128. Sigmoid activation is used for the inference layer, and all other layers use ReLU as the activation function.

Event mention detection model For word embeddings, we use the concatenation of a 300-dimensional pretrained GloVe embedding and the 50-dimensional embedding proposed by Turian et al. (2010). The character embedding layer is a combination of an 8-dimensional embedding layer and three 1D convolution layers with kernel sizes of 3, 4, 5 with 50 filters.

4.1.3 Evaluation Metrics

We follow the standard evaluation setup adopted in the official evaluation of the KBP event nugget detection and coreference task. This evaluation setup is based on four distinct scoring measures — MUC (Vilain et al., 1995) , B^3 (Bagga and Baldwin, 1998), $CEAF_e$ (Luo, 2005) and BLANC (Recasens and Hovy, 2011) — and the unweighted average of their F-scores (AVG-F). We use AVG-F as the main evaluation measure when comparing system performances.

4.2 Results

We present the experimental results on the KBP 2017 corpus in Table 3. In the following, we compare the performance of methods with different network architectures and experimental settings.

Comparison of network architectures We compare the results of the interactive inference network (Interact) with the biLSTM baseline model (biLSTM).

The biLSTM baseline model does not have the interaction layer. Instead, the last hidden vectors of the biLSTM layer are concatenated and fed into the inference layer directly.

When trained solely on the event coreference corpus (standard), the model with the interactive inference network performs slightly better than the biLSTM baseline model, as shown in rows 1 and 3. However, with an additional pretraining step of argument compatibility learning (transfer), the interact inference network outperforms the biLSTM baseline model by a considerable margin, as shown in rows 2 and 4. We conclude that the interactive inference network can better capture the complex interactions between two event mentions, accounting for the difference in performance.

Effect of transfer learning Regardless of the network structure, we observe a considerable improvement in performance by pretraining the model as an argument compatibility classifier. The biLSTM baseline model achieves an improvement of 1.25 points in AVG-F by doing transfer learning, as can be seen in rows 1 and 2. As for the interactive inference network, an improvement of 1.75 points in AVG-F is achieved, as can be seen in rows 3 and 4. These results provide suggestive evidence that our proposed transfer learning framework, which utilizes a large unlabeled corpus to perform argument compatibility learning, is effective.

Effect of iterative relabeling We achieve another boost in performance by using the trained event coreference resolver to relabel the training samples for argument compatibility learning. The best result is achieved after two iterations (row 5) with an improvement of 2.26 points in AVG-F compared to the standard interactive inference network (row 3). However, we are not able to obtain further gains with more iterations of relabeling (row 6). We speculate that the difference in event coreference model predictions across different iterations is not big enough to have a perceivable impact, but additional experiments are needed to determine the reason.

Type	Event Mention Pair	Gold	System
Explicit	m_1: ... the building where 13 people were **killed** will be razed, and a memorial ... m_2: In that case, George Hennard **killed** 23 people at a Luby 's restaurant, ...	non-coref	non-coref
Explicit	m_1: Ten relatives of the victims arrived at the airport Sunday before **traveling** to the city of Jiangshan. m_2: On Monday , the victims' relatives **went** to the Jiangshan Municipal Funeral Parlor.	non-coref	non-coref
Implicit	m_1: ... a young woman **protester** was brutally slapped while she was demonstrating ... m_2: ... explain why a women **protester** in her 60s was beaten up by policemen ...	non-coref	coref
Implicit	m_1: She died from a brain hemorrhage on July 10, 2003, ... m_2: ... has denied killing his second wife, whom he says **died** in a car accident.	non-coref	non-coref
Implicit	m_1: Nationwide demonstrations held in France to **protest** gay marriage. m_2: ... to **protest** against the country's plan to legalize same-sex marriage.	coref	coref
General	m_1: ... Connecticut elementary school **shooting** has reignited the debate over gun control. m_2: Gun supporters hold that people, not guns, are to blame for the **shootings**.	non-coref	coref
General	m_1: Industrial accidents have injured and **killed** Foxconn workers, and the company also experienced ... m_2: ... explosion in May 2011 at Foxconn 's Chengdu factory **killed** three workers ...	non-coref	non-coref

Table 4: Examples of event pairs with related triggers. Trigger words are boldfaced, and words with (in)compatibility information are colored in blue.

Comparison with the state of the art Comparing row 5 and 7, we can see that our method outperforms the previous state-of-the-art model (Jiang et al., 2017) by 1.42 points in AVG-F.

5 Discussion

In this section, we conduct a qualitative analysis of the outputs of our best-performing system (the Interact (transfer, 2nd iter) system in Table 3) on the event coreference dataset and the unseen event mention pairs extracted from the unlabeled corpus.

5.1 Compatibility Classification

We focus on the samples with related triggers having either compatible or incompatible arguments (Table 4). These samples can be roughly classified into the following categories:

Explicit argument compatibility The existence of identical/distinct time phrases, numbers, location names or person names in the context is the most explicit form of (in)compatibility.

For these event pairs, the existence of identical/distinct phrases with the same NER type is a direct clue toward deciding their coreferent status. Making use of this nature, we perform filtering on the set of compatible samples acquired from the unlabeled corpus in order to remove samples with explicit incompatibility.

Our model can recognize this type of (in)compatibility with a relatively high accuracy. Both examples shown in Table 4 are classified correctly.

Implicit argument compatibility Event pairs with implicit (in)compatible arguments require external knowledge to resolve.

We present three examples in Table 4. In the first example, the knowledge that a woman *in her 60s* is generally not referred to as being *young* is required to determine the incompatibility. Similarly, the knowledge that both *brain hemorrhage* and *car accident* are causes of people's death are required to classify the second example correctly.

While the performance on samples with implicit (in)compatibility is not as good as that on samples with explicit (in)compatibility, our system is able to capture implicit (in)compatibility to some extent. We believe that this type of (in)compatibility is difficult to be captured with the argument features that are designed based on the outputs of argument extractors and entity coreference resolvers, and that the ability to resolve implicit (in)compatibility contributes largely to our system's performance improvements.

General-specific incompatibilities Event mentions describing general events pose special challenges to the task of event coreference resolution.

In Table 4, we present two typical examples of this category. In the first example, the second event mention does not refer to any specific shooting event in the real world, in contrast to the first event mention, which describes a specific school shooting event. Similarly for the second example, where the first event mention depicts a general event and the second event mention depicts a specific one.

General event mentions typically have few or even no arguments and modifiers, making the identification of non-coreference relations very challenging. Since we cannot rely on argument compatibility, a deeper understanding of the semantics of the event mentions is needed. General

	Event Mention Pair	Type	System
I	m_1: What would have happened if Steve Jobs had never **left** Apple ...	-	-
	m_2^a: ...in the state that is today if John hadn't **left**.	Explicit	non-coref
	m_2^b: ...in the state that is today if she hadn't **left**.	Implicit	non-coref
	m_2^c: ...in the state that is today if he hadn't **left**.	Implicit	coref
II	m_1: Police arrest 6 men for **gangraping** housewife in northern India.	-	-
	m_2^a: Indian police have arrested six men for allegedly **gangraping** a 29-year-old housewife ...	Explicit	coref
	m_2^b: Indian police have arrested six men for allegedly **gangraping** a woman ...	Implicit	coref
	m_2^c: Indian police have arrested six men for allegedly **gangraping** a medical student ...	Implicit	non-coref
III	m_1: Nationwide demonstrations in France to **protest** gay marriage.	-	-
	m_2^a: ...took to the streets across the country to **protest** against the country's plan to legalize same-sex marriage.	Implicit	coref
	m_2^b: ...took to the streets across the country to **protest** against the contentious citizenship amendment bill.	Implicit	non-coref

Table 5: Case study on manually-generated event mention pairs. Trigger words are boldfaced, and the target arguments are colored in blue.

event mentions account for a considerable fraction of our system's error, since they are quite pervasive in both news articles and discussion forum threads.

5.2 Case Study

To better understand the behavior of our system, we perform a case study on manually-generated event pairs. Specifically, for a given pair of event mentions, we first alter only one of the arguments and keep the rest of the content fixed. We then observe the behavior of the system across different variations of the altered argument (Table 5).

Example I In this example, we pick the AGENT-argument as the target and alter the AGENT-argument of the second event mention. The event pair (m_1, m_2^a) is non-coreferent due to the explicit incompatibility between *Steve Jobs* and *John*, and the system's prediction is also non-coreferent. Further, we alter the target argument to the pronoun *she* (m_2^b), resulting in an implicit incompatibility in the AGENT argument since the *Steve Jobs* is generally not considered a feminine name. As expected, the system classifies the event pair (m_1, m_2^b) as non-coreferent. Finally, when we alter the target argument to *he* (m_2^c), the system correctly classifies the resulting pair as coreferent.

Example II In this example, we pick the PATIENT-argument as the target and alter the PATIENT-argument of the second event mention. The system classifies the event pair (m_1, m_2^a) as coreferent, which is reasonable considering the presence of the explicit compatible arguments *housewife* and *29-year-old housewife*. Further, when we alter the target argument to *woman* (m_2^b), the system output is still coreferent. This is consistent with our prediction: the event mentions are likely to be coreferent judging only from the context of the two event mentions. However, when we alter the target argument to *medical student* (m_2^c), the event pair would become non-coreferent due to the incompatibility between *medical student* and *housewife*. The system classifies the event pair correctly.

Example III In this example, we pick the REASON-argument as the target and alter the REASON-argument of the second event mention. The event pair (m_1, m_2^a) has a pair of implicit compatible arguments in the REASON-argument role and is likely to be coreferent. In contrast, altering the target argument to *contentious citizenship amendment bill* (m_2^b) would yield an pair of implicit incompatible arguments, and the resulting event pair would become non-coreferent. Our system classifies both event pairs correctly.

6 Conclusion

We proposed an iterative transfer learning framework for event coreference resolution. Our method exploited a large unlabeled corpus to learn a wide range of (in)compatibilities between arguments, which contributes to the improvement in performance on the event coreference resolution task. We achieved state-of-the-art results on the KBP 2017 English event coreference dataset, outperforming the previous state-of-the-art system. In addition, a qualitative analysis of the system output confirmed the ability of our system to capture (in)compatibilities between two event mentions.

Acknowledgments

We thank the three anonymous reviewers for their detailed comments on an earlier draft of the paper. This work was supported in part by NSF Grants IIS-1219142 and IIS-1528037 and JST CREST Grant Number JPMJCR1301, Japan.

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 9th 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 LREC Workshop on Linguistics Coreference*, pages 563–566.

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

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. 2014. SinoCoreferencer: An end-to-end Chinese event coreference resolver. In *Proceedings of the 9th International Conference on Language Resources and Evaluation*, pages 4532–4538.

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 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.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading Wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1870–1879.

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

Zheng Chen, Heng Ji, and Robert Haralick. 2009. A pairwise event coreference model, feature impact and evaluation for event coreference resolution. In *Proceedings of the RANLP Workshop on Events in Emerging Text Types*, 3, pages 17–22.

Prafulla Kumar Choubey and Ruihong Huang. 2017. Event coreference resolution by iteratively unfolding inter-dependencies among events. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2124–2133.

Agata Cybulska and Piek Vossen. 2013. Semantic relations between events and their time, locations and participants for event coreference resolution. In *Proceedings of the 9th International Conference on Recent Advances in Natural Language Processing*, pages 156–163.

Joe Ellis, Jeremy Getman, Dana Fore, Neil Kuster, Zhiyi Song, Ann Bies, and Stephanie Strassel. 2015. Overview of linguistic resources for the TAC KBP 2015 evaluations: Methodologies and results. In *Proceedings of the TAC KBP 2015 Workshop*.

Joe Ellis, Jeremy Getman, Neil Kuster, Zhiyi Song, Ann Bies, and Stephanie Strassel. 2016. Overview of linguistic resources for the TAC KBP 2016 evaluations: Methodologies and results. In *Proceedings of the TAC KBP 2016 Workshop*.

Jeremy Getman, Joe Ellis, Zhiyi Song, Jennifer Tracey, and Stephanie Strassel. 2017. Overview of linguistic resources for the TAC KBP 2017 evaluations: Methodologies and results. In *Proceedings of the TAC KBP 2017 Workshop*.

Yichen Gong, Heng Luo, and Jian Zhang. 2018. Natural language inference over interaction space. In *Proceedings of the 6th International Conference on Learning Representations*.

Sepp Hochreiter and Jüngen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Shanshan Jiang, Yihan Li, Tianyi Qin, Qian Meng, and Bin Dong. 2017. SRCB entity discovery and linking (EDL) and event nugget systems for TAC 2017. In *Proceedings of the TAC KBP 2017 Workshop*.

Sebastian Krause, Feiyu Xu, Hans Uszkoreit, and Dirk Weissenborn. 2016. Event Linking with Sentential Features from Convolutional Neural Networks. In *Proceedings of the 20th 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 9th International Conference on Language Resources and Evaluation*, pages 4539–4544.

Jing Lu and Vincent Ng. 2016. Event coreference resolution with multi-pass sieves. In *Proceedings of the 10th International Conference on Language Resources and Evaluation*, pages 3996–4003.

Jing Lu and Vincent Ng. 2017. Joint learning for event coreference resolution. In *Proceedings of 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 90–101.

Jing Lu and Vincent Ng. 2018. Event coreference resolution: A survey of two decades of research. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pages 5479–5486.

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: Technical Papers*, pages 3264–3275.

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

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of the 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.

Vincent Ng and Claire Cardie. 2002. Improving machine learning approaches to coreference resolution. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 104–111.

Boyuan Pan, Yazheng Yang, Zhou Zhao, Yueting Zhuang, Deng Cai, and Xiaofei He. 2018. Discourse marker augmented network with reinforcement learning for natural language inference. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 989–999.

Robert Parker, David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. 2009. English Gigaword fourth edition. In *Linguistic Data Consortium*.

Haoruo Peng, Yangqi 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.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1532–1543.

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

Satyan Sangeetha and Michael Arock. 2012. Event coreference resolution using mincut based graph clustering. In *Proceedings of the 4th International Workshop on Computer Networks & Communications*, pages 253–260.

Wee Meng Soon, Hwee Tou Ng, and Daniel Chung Yong Lim. 2001. A machine learning approach to coreference resolution of noun phrases. *Computational Linguistics*, 27(4):521–544.

Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: A simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 384–394.

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.

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.

Zhilin Yang, Bhuwan Dhingra, Ye Yuan, Junjie Hu, William W. Cohen, and Ruslan Salakhutdinov. 2017. Words or Characters? Fine-grained gating for reading comprehension. In *Proceedings of the 5th International Conference on Learning Representations*.