Graph-Based Decoding for Event Coreference and Sequencing Resolution

Zhengzhong Liu and **Teruko Mitamura** and **Eduard Hovy**
Language Technologies Institute
Carnegie Mellon University
Pittsburgh, PA 15213, USA
{liu, teruko, hovy}@cs.cmu.edu

Abstract

Events in text documents are interrelated in complex ways. In this paper, we study two types of relation: **Event Coreference** and **Event Sequencing**. We show that the popular tree-like decoding structure for automated Event Coreference is not suitable for Event Sequencing. To this end, we propose a graph-based decoding algorithm that is applicable to both tasks. The new decoding algorithm supports flexible feature sets for both tasks. Empirically, our event coreference system has achieved state-of-the-art performance on the TAC-KBP 2015 event coreference task and our event sequencing system beats a strong temporal-based, oracle-informed baseline. We discuss the challenges of studying these event relations.

Title and Abstract in Chinese

基于图的事件共指消解以及依序关系解码算法

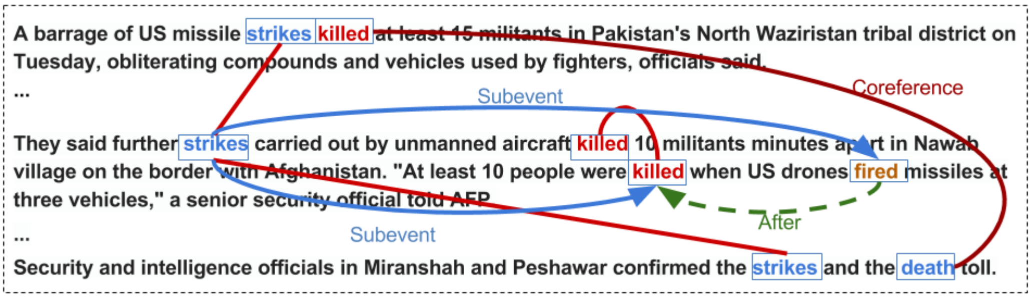
文本中提及的事件之间常存在着复杂的关系。在这篇文章中，我们主要研究两种关系：事件间的共指关系以及依序关系。我们指出，常被应用在事件指代消歧上的传统的树结构解码方式并不适用于解决事件依序问题。为了解决这个问题，我们提出了一种适用于两种情况的新的图结构解码算法。这个算法让我们可以为这两个任务设计灵活的特征。在实验上，我们的事件共指消解系统在TAC-KBP 2015的共指消解任务上达到了目前最佳的水平。同时，我们的依序关系系统的结果超过了一个基于时间分析并有部分正确答案的基线系统。最后我们分析了结果，并讨论了事件关系判别任务的一些挑战。

1 Introduction

Events are important building blocks of documents. They play a key role in document understanding tasks, such as information extraction (Chambers and Jurafsky, 2011), news summarization (Vossen and Caselli, 2015), story understanding (Mostafazadeh et al., 2016). Conceptually, **events** correspond to state changes and normally include a location, a time interval, and several entities/participants. In a text, events are realized as text spans, normally as verbs and nouns that indicate state changes (Vendler, 1957). The text spans are often referred to as **event mentions** or **event nuggets** (We use the term event mention in this paper.). The textual mentions of events have rich relations among them, and collectively convey the meaning of one or more related documents. In this paper, we study two different types of relation: **Event Hopper Coreference (EH)** and **Event Sequencing (ES)**.

Event Coreference: There is a rich literature on the Event Coreference problem (Liu et al., 2014; Cybulska and Vossen, 2014; Lu et al., 2016; Peng et al., 2016; Lu and Ng, 2017; Araki, 2018). By analogy to entity coreference, the "same" conceptual event may be realized by multiple text spans (event mentions). The coreference problem aims at identifying these relations to recover events from the text spans. The **Event Hopper** Coreference task in the TAC-KBP evaluation campaign defines coreference links as follows (Mitamura et al., 2018): Two event mentions are considered coreferent if they refer to the conceptually same underlying event, even if their arguments are not strictly identical. For example,

Proceedings of the 27th International Conference on Computational Linguistics, pages 3645–3657
Santa Fe, New Mexico, USA, August 20-26, 2018.



A barrage of US missile strikes killed at least 15 militants in Pakistan's North Waziristan tribal district on Tuesday, obliterating compounds and vehicles used by fighters, officials said.

...

They said further strikes carried out by unmanned aircraft killed 10 militants minutes apart in Nawab village on the border with Afghanistan. "At least 10 people were killed when US drones fired missiles at three vehicles," a senior security official told AFP.

...

Security and intelligence officials in Miranshah and Peshawar confirmed the strikes and the death toll.

Figure 1: Example of Event Coreference and Sequence relations. Red lines are coreference links; solid blue arrows represent Subevent relations; dotted green arrows represent After relations.

mentions that share similar temporal and location scope, though not necessarily the same expression, are considered to be coreferent (*Attack in Baghdad on Thursday* vs. *Bombing in the Green Zone last week*). This means that the event arguments of coreferential events mentions can be non-coreferential (18 killed vs. dozens killed), as long as they refer to the same event, judging from the available evidence.

Event Sequencing: The coreference relations build up events from scattered mentions. On the basis of events, various other types of relations can then be established between them. The Event Sequencing task studies one such relation. The task is motivated by Schank's *scripts* (Schank and Abelson, 1977), which suggests that human organize information through procedural data structures, reassembling sequences of events. For example, the list of verbs *order, eat, pay, leave* may trigger the restaurant script. A human can conduct reasoning with a typical ordering of these events based on common sense (e.g., *order* should be the first event, *leave* should be the last event).

The ES task studies how to group and order events from text documents belonging to the same script. Figure 1 shows some annotation examples. Conceptually, event sequencing relations hold between the events, while coreference relations hold between textual event mentions. Given a document, the ES task requires systems to identify events within the same script and classify their inter-relations. These relations can be represented as labeled Directed Acyclic Graphs (DAGs). There are two types of relations[1]: **After** relations connect events following script orders (e.g. *order* followed by *eating*); **Subevent** relations connect events to a larger event that contains them. In this paper, we focus only on the **After** relations.

Since script-based understanding is built in the ES task, it has some unique properties comparing to pure temporal ordering: 1) event sequences from different scripts provide separate logical divisions of text, while temporal ordering considers all events to lie on a single timeline; 2) temporal relations for events occurring at similar time points may be complicated. Script-based relations may alleviate the problem. For example, if a `bombing` `kills` some people, the temporal relation of the `bombing` and `kill` may be "inclusion" or "after". This is considered an **After** relation in ES because `bombing` causes the `kill`ing.

For structure prediction, decoding — recovering the complex structure from local decisions — is one of the core problems. The most successful decoding algorithm for coreference nowadays is mention ranking based (Björkelund and Kuhn, 2014; Durrett and Klein, 2014; Lee et al., 2017). These models rank the antecedents (mentions that appear earlier in discourse) and recover the full coreference clusters from local decisions. However, unlike coreference relations, sequencing relations are directed. Coreference decoding algorithms cannot be directly applied to such relations (§3.1). To solve this problem, we propose a unified graph-based framework that tackles both event coreference and event sequencing. Our method achieves state-of-the-art results on the event coreference task (§4.4) and beats an informed baseline on the event sequencing task (§4.5). Finally, we analyze the results and discuss the difficult challenges for both tasks (§5). Detailed definitions of these tasks can be found in the corresponding task documents[2].

2 Related Work

Many researchers have worked on event coreference tasks since Humphreys et al. (1997). Recent advances in event coreference have been promoted by the availability of annotated corpora. However, due to

[1] Detailed definition of relations can be found in http://cairo.lti.cs.cmu.edu/kbp/2016/after/

[2] `http://cairo.lti.cs.cmu.edu/kbp/2017/event/documents`

the complex nature of events, approaches to event coreference adopt quite different assumptions and definitions. Most of event coreference researches are conducted on the popular ACE corpus (Chen and Ji, 2009; Chen et al., 2009; Sangeetha and Arock, 2012; Chen and Ng, 2013; Chen and Ng, 2015). Unlike the TAC KBP setting, the definition of event coreference in the ACE corpus requires strict argument matching. Work on the Intelligence Community (IC) Corpus (Hovy et al., 2013; Cybulska and Vossen, 2012; Liu et al., 2014; Araki et al., 2014) considers event relations on a restricted domain (i.e., terrorist events). Works on the ECB corpus (Lee et al., 2012; Cybulska and Vossen, 2014) focuses on both within-document and cross-document coreference.

Our work follows the line of work promoted by the TAC-KBP event nugget tasks (Mitamura et al., 2015). There is a small but growing amount of work on conducting event coreference on the TAC-KBP datasets (Lu et al., 2016; Peng et al., 2016; Lu and Ng, 2017). The TAC dataset uses a relaxed coreference definition comparing to other corpora, requiring two event mentions to intuitively refer to the same real-world event despite differences of their participants.

For event sequencing, there are few supervised methods on script-like relation classification due to the lack of data. To the best of our knowledge, the only work in this direction is by Araki et al. (2014). This work focuses on the other type of relations in the event sequencing task: **Subevent** relations. There is also a rich literature on unsupervised script induction (Chambers and Jurafsky, 2008; Cheung et al., 2013; Rudinger et al., 2015; Pichotta and Mooney, 2016; Ferraro and Durme, 2016) that extracts scripts as a type of common-sense knowledge from raw documents. The focus of this work is to make use of massive collections of text documents to mine event co-occurrence patterns. In contrast, our work focuses on parsing the detailed relations between event mentions in each document.

Another line of work closely related to event sequencing is to detect other temporal relations between events. Recent computational approaches for temporal detection are mainly conducted on the TimeBank corpus (Pustejovsky et al., 2002). There have been several studies on building automatic temporal reasoning systems (Uzzaman and Allen, 2010; Do et al., 2012; Chambers et al., 2014). In comparison, the Event Sequencing task is motivated by the Script theory, which places more emphasis on common-sense knowledge about event chronology.

3 Model

3.1 Graph-Based Decoding Model

In the Latent Antecedent Tree (LAT) model popularly used for entity coreference decoding (Fernandes et al., 2012; Björkelund and Kuhn, 2014), each node represents an event mention and each arc a coreference relation, and new mentions are connected to some past mention considered most similar. Thus the LAT model represents the decoding structure as a tree. This can represent any coreference cluster, because coreference relations are by definition equivalence relations[3].

In contrast, tree structures cannot always fully cover an Event Sequence relation graph, because 1) the After links are directed, not symmetric, and 2) multiple event nodes can link to one node, resulting in multiple parents.

To solve this problem, we extend the LAT model and propose its graph version, namely the Latent Antecedent Graph (LAG) model. Figure 2 contrast LAT and LAG with decoding examples. The left box shows two example decoded trees in LAT, where each node has one single parent. The right box shows two example decoded trees in LAG, where each node can be linked to multiple parents.

Formally, we define the series of (pre-extracted) event mentions of the document as $M = \{m_0, m_1, ..., m_n\}$, following their discourse order. m_0 is an artificial root node preceding all mentions. For each mention m_j, let A_j be the set of its potential antecedents: $A_j = \{m_0, m_1, ..., m_{j-1}\}$. Let $\mathcal{A}$ denotes the set of antecedents for all the mentions in the sequence $\{A_0, A_1, ..., A_n\}$. The two tasks in question can be considered as finding the appropriate antecedent(s) from $\mathcal{A}$. Similarly, we define the gold antecedent set $\tilde{\mathcal{A}} = \{\tilde{A}_0, \tilde{A}_1, ..., \tilde{A}_n\}$, where $\tilde{A}_i$ represent the set of antecedents of m_i allowed by the gold standard. In the coreference task, $\tilde{A}_i$ contains all antecedents that are coreferent with m_i. In the sequencing task, $\tilde{A}_i$ contains all antecedents that have an $After$ relation to m_i.

[3] An equivalence relation is reflexive, symmetric and transitive.

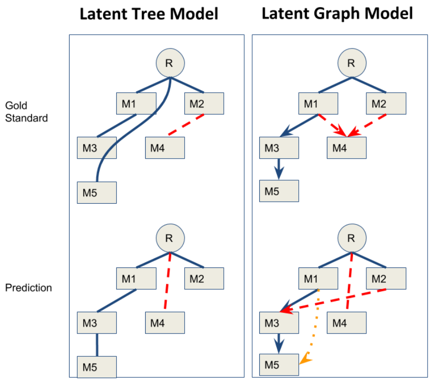


Figure 2: Latent Tree Model (left): tree structure formed by undirected links. Latent Graph Model (right): a DAG form by directed links. Dashed red links highlight the discrepancy between prediction and gold standard. The dotted yellow link (bottom right) can be inferred from other links.

We can now describe the decoding process. We represent each arc as $\langle m_i, m_j, r\rangle (i < j)$, where r is the relation name. The relation direction can be specified in the relation name r (e.g. r can be *after.forward* or *after.backward*). Further, an arc from the root node m_0 to node m_j represents that m_j does not have any antecedent. The score of the arc is the dot product between the weight parameter $\vec{w}$ and a feature vector $\Phi(\langle m_i, m_j, r\rangle)$, where Φ is an arc-wise feature function. The decoded graph z can be determined by a set of binary variables $\vec{z}$, where $\vec{z}_{ijr} = 1$ if there is an arc $\langle m_i, m_j, r\rangle$ or 0 otherwise. The final score of z is the sum of scores of all arcs:

$$score(z) = \sum_{i,j,r} \vec{z}_{ijr} \vec{w} \cdot \Phi(\langle m_i, m_j, r\rangle) \tag{1}$$

The decoding step is to find the output $\hat{z}$ that maximizes the scoring function:

$$\hat{z} = \arg \max_{z \in \mathcal{Z}(\mathcal{A})} score(z) \tag{2}$$

where $\mathcal{Z}(\mathcal{A})$ denotes all possible decoding structures given the antecedent sets $\mathcal{A}$. It is useful to note that the decoding step can be applied in the same way to the gold antecedent set $\tilde{\mathcal{A}}$.

Algorithm 1 shows the Passive-Aggressive training algorithm (Crammer et al., 2006) used in our decoding framework. Line 8 decodes the maximum scored structure from all possible gold standard structures using the current parameters $\vec{w}$. Intuitively, this step tries to find **the "easiest" correct graph** — the correct graph with the highest score — for the current model. Several important components remain unspecified in algorithm 1: (1) the decoding step (line 6, 8); (2) the match criteria: whether to consider the system decoding structure as correct (line 7); (3) feature delta: computation of feature difference (line 9); (4) loss computation (line 10). We detail the actual implementation of these steps in §3.1.2.

3.1.1 Minimum Decoding Structure

Similar to the LAT model, there may be many decoding structures representing the same configuration. In LAT, since there is exactly one link per node, the number of links in different decoding structures is the same, hence comparable. In LAG, however, one node is allowed to link to multiple antecedents, creating a potential problem for decoding. For example, consider the sequence $m_1 \xrightarrow{\text{after}} m_2 \xrightarrow{\text{after}} m_3$, both of the following structures are correct:

1. $\langle m_1, m_2, after\rangle$, $\langle m_2, m_3, after\rangle$
2. $\langle m_1, m_2, after\rangle$, $\langle m_2, m_3, after\rangle$, $\langle m_1, m_3, after\rangle$

However, the last relation in the second decoding structure can actually be inferred via transitivity. We do not intend to spend the modeling power on such cases. We empirically avoid such redundant cases by using the **transitive reduction graph** for each structure. For a directed acyclic graph, a transitive

Algorithm 1: PA algorithm for training

```
1  **Input:** Training data D, number of iterations T
2  **Output:** Weight vector $\vec{w}$
3  $\vec{w} = \vec{0}$;
4  $\langle \mathcal{A}, \tilde{\mathcal{A}} \rangle \in D$;
5  **for** $t \leftarrow 1..T$ **do**
6      $\hat{z} = \arg\max_{\mathcal{Z}(\mathcal{A})} score(z)$;
7      **if** $\neg Match(\hat{z}, \tilde{\mathcal{A}})$ **then**
8          $\tilde{z} = \arg\max_{\mathcal{Z}(\tilde{\mathcal{A}})} score(z)$;
9          $\Delta = FeatureDelta(\tilde{z}, \hat{z})$;
10         $\tau = \frac{loss(\tilde{z},\hat{z})}{||\Delta||^2}$;
11         $w = w + \tau\Delta$;
12 **return** w;
```

reduction graph contains the fewest possible edges that have the same reachability relation as the original graph. In the example above, structure 1 is a transitive reduction graph for structure 2. We call the decoding structures that corresponding to the reduction graphs as *minimum decoding structures*. For LAG, we further restrict $\mathcal{Z}(\mathcal{A})$ to contain only minimum decoding structures.

3.1.2 Training Details in Latent Antecedent Graph

In this section, we describe the decoding details for LAG. Note that if we enforce a single antecedent for each node (as in our coreference model), it falls back to the LAT model (Björkelund and Kuhn, 2014).

Decoding: We use a greedy **best-first decoder** (Ng and Cardie, 2002), which makes a left-to-right pass over the mentions. The decoding step is the same for line 6 and 8. The only difference is that we will use gold antecedent set ($\tilde{\mathcal{A}}$) at line 8. For each node m_j, we keep all links that score higher than the root link $\langle 0, m_j, r \rangle$.

Cycle and Structure Check: Incremental decoding a DAG may introduce cycles to the graph, or violate the minimum decoding structure criterion. To solve this, we maintain a set $R(m_i)$ that is reachable from m_i during the decoding process. We reject a new link ($\langle m_j, m_i \rangle$ if $m_j \in R(m_i)$) to avoid cycles. We also reject a redundant link ($\langle m_i, m_j \rangle$ if $m_j \in R(m_i)$) to keep a minimum decoding structure. Our current implementation is greedy, we leave investigations of search or global inference based algorithms to future work.

Selecting the Latent Event Mention Graph: Note that sequence relations are on the event level. Given a unique event graph, it may still correspond to multiple mention graphs. In our implementation, we use a minimum set of event mentions to represent the full event graph by taking one single mention from each event. Following the "easiest" intuition, we select the single mention that will result in the highest score given the current feature weight w.

Match Criteria: We consider two graphs to match when their inferred graphs are the same. The inferred graph is defined by taking the transitive closure of the graph and propagate the links through the coreference relations. For example, in Figure 1, the mention `fired` will be linked to two `killed` mentions after propagation.

Feature Delta: In structural perceptron training (Collins, 2002), the weights are updated directly by the feature delta. For all the features $\tilde{f}$ of the gold standard graph $\tilde{z}$ and features $\hat{f}$ of a decoded graph $\hat{z}$, the feature delta is simply: $\Delta = \tilde{f} - \hat{f}$. However, a decoded graph may contain links that are not directly presented but inferable from the gold standard graph. For example, in Figure 2, the prediction graph has a link from $M5$ to $M1$ (the orange arc), which is absent but inferable from the gold standard tree. If we keep these links when computing Δ, the model does not converge well. We thus remove the features on the inferable links from $\hat{f}$ when computing Δ.

Loss: We define the loss to be the number of different edges in two graphs. Following Björkelund and Kuhn (2014), we further penalize erroneous root attachment: an incorrect link to the root m_0 adds the loss

Head	Headword token and lemma pair, and whether they are the same.
Type	The pair of event types, and whether they are the same.
Realis	The pair of realis types and whether they are the same.
POS	POS pair of the two mentions and whether they are the same.
Exact Match	Whether the 5-word windows of the two mentions matches exactly.
Distance	Sentence distance between the two mentions.
Frame	Frame name pair of the two mentions and whether they are the same.
Syntactic	Whether a mention is the syntactic ancestor of another.

Table 1: Coreference Features. Parsing is done using Stanford CoreNLP (Manning et al., 2014); frame names are produced by Semafor (Das and Smith, 2011).

by 2. For example, in Figure 2 the prediction graph (bottom right) incorrectly links m_4 to Root and misses a link to m_3, which cause a total loss of 3. In addition, to be consistent with the feature delta computation, we do not compute loss for predicted links that are inferable from the gold standard.

3.2 Features

3.2.1 Event Coreference Features

For event coreference, we design a simple feature set to capture syntactic and semantic similarity of arcs. The main features are summarized in Table 1. In the TAC KBP 2015 coreference task setting, the event mentions are annotated with two attributes. There are 38 event types and subtype pairs (e.g., *Busness.Merge-Org, Conflict.Attack*). There also 3 realis type: events that actually occurred are marked as *Actual*; events that are not specific are marked as *Generic*; other events such as future events are marked as *Other*. For these two attributes, we use the gold annotations in our feature sets.

3.2.2 Event Sequencing Features

An event sequencing system needs to determine whether the events are in the same script and order them. We design separate feature sets to capture these aspects: the Script Compatibility set considers whether mentions should belong to the same script; the Event Ordering set determines the relative ordering of the mentions. Our final features are the cross products of features from the following 3 sets.

1. **Surface-Based Script Compatibility**: these features capture whether two mentions are script compatible based on the surface information, including:
   - Mention headword pair.
   - Event type pair.
   - Whether two event mentions appear in the same cluster in Chambers's event schema database (Chambers and Jurafsky, 2010).
   - Whether the two event mentions share arguments, and the semantic frame name of the shared argument (produced by the Semafor parser (Das and Smith, 2011)).
2. **Discourse-Based Script Compatibility**: these features capture whether two event mentions are related given the discourse context.
   - Dependency path between the two mentions.
   - Function words (words other than Noun, Verb, Adjective and Adverb) in between the two mentions.
   - The types of other event mentions between the two mentions.
   - The sentence distance of two event mentions.
   - Whether there are temporal expressions (AGM-TMP slot from a semantic parser (Tratz and Hovy, 2011)) in the sentences of the two mentions.
3. **Event Ordering**: this feature set tries to capture the ordering of events. We use the discourse ordering of two mentions (forward: the antecedent is the parent; backward: the antecedent is the child), and temporal ordering produced by Caevo (Chambers et al., 2014).

Taking the *after* arc from `fired` to `killed` in Figure 1 as an example, a feature after the cross product is: Event type pair is *Conflict.Attack* and *Life.Die*, discourse ordering is *backward*, and sentence distance is 0.

4 Experiments

4.1 Dataset

We conduct experiments on the dataset released in Text Analysis Coreference (TAC-KBP) 2017 Event Sequencing task (released by LDC under the catalog name LDC2016E130). This dataset contains rich event relation annotations, with event mentions and coreference annotated in TAC-KBP 2015, and additional annotations on Event Sequencing[4]. There are 158 documents in the training set and 202 in the test set, selected from general news articles and forum discussion threads. The event mentions are annotated with 38 type-subtype and 3 realis status (Actual, Generic, Other). Event Hopper, After, and Subevent links are annotated between event mentions. For all experiments, we develop our system and conduct ablation studies using 5-fold cross-validation on the training set, and report performance on the test set.

4.2 Baselines and Benchmarks

Coreference: we compare our event coreference system against the top performing systems from TAC-KBP 2015 (LCC, UI-CCG, and LTI). In addition, we also compare the results against two official baselines (Mitamura et al., 2015): the Singleton baseline that put each event mention in its own cluster and the Match baseline that creates clusters based on mention type and realis status match.

Sequencing: This work is an initial attempt to this problem, so there is currently no comparable prior work on the same task. We instead compare with a baseline using event temporal ordering systems. We use a state-of-the-art temporal system named Caevo (Chambers et al., 2014). To make a fair comparison, we feed the gold standard event mentions to the system along with mentions predicted by Caevo[5]. However, since the script-style After links are only connected between mentions in the same script, directly using the output of Caevo produces very low precision. Instead, we run a stronger baseline: we take the gold standard script clusters and then only ask Caevo to predict links within these clusters (Oracle Cluster + Temporal).

4.3 Evaluation Metrics

Evaluating Event Coreference: We evaluate our results using the official scorer provided by TAC-KBP, which uses 4 coreference metrics: *BLANC* (Recasens and Hovy, 2011), *MUC* (Chinchor, 1992), B^3 (Bagga and Baldwin, 1998) and *CEAF-E* (Luo, 2005). Following the TAC KBP task, systems are ranked using the average of these 4 metrics.

Evaluating Event Sequencing: The TAC KBP scorer evaluates event sequencing using the metric of the TempEval task (UzZaman, 2012; UzZaman et al., 2013). The TempEval metric calculates special precision and recall values based on the closure and reduction graphs:

$$Precision = \frac{|Response^- \cap Reference^+|}{|Response^-|} \quad Recall = \frac{|Reference^- \cap Response^+|}{|Reference^-|}$$

where $Response$ represents the After link graph from the system response and $Reference$ represents the After link graph from the gold standard. G^+ represents the graph closure for graph G and G^- represents the graph reduction for graph G. As preprocessing, relations are automatically propagated through coreference clusters (currently using gold standard clusters). The final score is the standard F-score: geometric mean of the precision and recall values.

[4] http://cairo.lti.cs.cmu.edu/kbp/2016/after/

[5] We keep the mentions predicted by Caevo because its inference may be affected by these mentions.

	B^3	CEAF-E	MUC	BLANC	AVG.
`Singleton`	78.10	68.98	0.00	48.88	52.01
`Matching`	78.40	65.82	**69.83**	76.29	71.94
LCC	82.85	74.66	68.50	**77.61**	75.69
UI-CCG	83.75	75.81	63.78	73.99	74.28
LTI	82.27	75.15	60.93	71.57	72.60
This work	**85.59**	**79.65**	67.81	77.37	**77.61**

Table 2: Test Results for Event Coreference with the `Singleton` and `Matching` baselines.

	B^3	CEAF-E	MUC	BLANC	AVG.
ALL	81.97	74.80	76.33	76.07	77.29
-Distance	81.92	74.48	76.02	77.55	77.50
-Frame	82.14	75.01	76.28	77.74	77.79
-Syntactic	81.87	74.89	75.79	76.22	77.19

Table 3: Ablation study for Event Coreference.

4.4 Evaluation Results for Event Coreference

The test performance on Event Coreference is summarized in Table 2. Comparing to the top 3 coreference systems in TAC-KBP 2015, we outperform the best system by about 2 points absolute F-score on average. Our system is also competitive on individual metrics. Our model performs the best based on B^3 and CEAF-E, and is comparable to the top performing systems on MUC and BLANC.

Note that while the `Matching` baseline only links event mentions based on event type and realis status, it is very competitive and performs close to the top systems. This is not surprising since these two attributes are based on the gold standard. To take a closer look, we conduct an ablation study by removing the simple match features one by one. The results are summarized in Table 3. We observe that some features produce mixed results on different metrics: they provide improvements on some metrics but not all. This is partially caused by the different characteristics of different metrics. On the other hand, these features (parsing and frames) are automatically predicted, which make them less stable. Furthermore, the Frame features contain duplicate information to event types, which makes it less useful in this setting.

Besides the presented features, we have also designed features using event argument. However, we do not report the results since the argument features decrease the performance on all metrics.

4.5 Evaluation Results for Event Sequencing

The evaluation results on Event Sequencing is summarized in Table 4. Because the baseline system has access to the oracle script clusters, it produces high precision. However, the low recall value shows that it fails to produce enough After links. Our analysis shows that a lot of After relations are not indicated by clear temporal clues, but can only be solved with script knowledge. In Example 3, the baseline system is able to identify "fled" is after "ousted" from explicit marker "after". However, it fails to identify that "extradited" is after "arrested", which requires knowledge about prototypical event sequences.

(3) Eight months after the [$_{\text{transport}}$ fled] Ivory Coast when Gbagbo, the former president, was [$_{\text{End.Position}}$ ousted] by the French military. Blé Goudé was subsequently [$_{\text{Jail}}$ arrested] in Ghana and [$_{\text{transport}}$ extradited] Megrahi,[$_{\text{Jail}}$ jailed] for [$_{\text{Attack}}$ killing] 270 people in 1988. [6]

In our error analysis, we noticed that our system produces a large number of relations due to coreference propagation. One single wrong prediction can cause the error to propagate.

Besides memorizing the mention pairs, our model also tries to capture script compatibility through discourse signals. To further understand how much these signals help, we conduct an ablation study of the features in the discoursed based compatibility features (see §3.2.2). Similarly, we remove each feature group from the full feature set one by one and observe the performance change.

[6] The small red text indicates the event type for each mention.

	Prec.	Recall	F-Score
Oracle Cluster+Temporal	**46.21**	8.72	14.68
Our Model	18.28	**16.91**	**17.57**

Table 4: Test Results for event sequencing. The Oracle Cluster+Temporal system is using Caevo's result on the Oracle Clusters.

	Prec.	Recall	F-Score	Δ
Full	37.92	36.79	36.36	
- Mention Type	32.78	29.81	30.07	6.29
- Sentence	33.90	30.75	31.00	5.36
- Temporal	37.21	36.53	35.81	0.55
- Dependency	38.18	36.44	36.23	0.13
- Function words	38.08	36.51	36.18	0.18

Table 5: Ablation Study for Event Sequencing.

The results are reported in Table 5. While most of the features only affect the performance by less than 1 absolute F1 score, the feature sets after removing *mention* or *sentences* show a significant drop in both precision and recall. This shows that discourse proximity is the most significant ones among these features. In addition, the *mention* feature set captures the following *explain away* intuition: the event mentions A and B are less likely to be related if there are similar mentions in between. One such example can be seen in Figure 1, the event mention `fired` is more likely to relate to the closest `killed`, instead of the other `killed` in the first paragraph.

In addition, our performance on the development set is higher than the test set. Further analysis reveals two causes: 1) the coreference propagation step causes the scores to be very unstable, 2) our model only learns limited common sense ordering based on lexical pairs, which overfit to the small training corpus. Since the annotation is difficult to scale, it is important to use methods to harvest script common sense knowledge automatically, as in the script induction work (Chambers and Jurafsky, 2008).

5 Discussion

5.1 Event Coreference Challenges

Although we have achieved good performance on event coreference, upon closer investigation we found that most of the coreference decisions are still made based on simple word/lemma matching (note that the type and realis baseline is as high as 0.72 F1 score). The system exploits little semantic information to resolve difficult event coreference problems. A major challenge is that our system is not capable of utilizing event arguments: in fact, Hasler and Orasan (2009) found that only around 20% of the arguments in the same event slot are actually coreferent for coreferential event pairs in the ACE 2005 corpus. Furthermore, the TAC-KBP corpus uses a relaxed participant identity requirement for event coreference, which makes argument-based matching more difficult.

5.2 Event Sequencing Challenges

Our event sequencing performance is still low despite the introduction of many features. This task is inherently difficult because it requires a system to solve both the script clustering and event ordering tasks. The former task requires both common-sense knowledge and discourse reasoning. Reasoning is more important for long-term links since there are no explicit clues like prepositions and dependencies to be exploited. The ablation study shows that discourse features like sentence distance are more effective, which indicates that our model mainly relies on surface clues and has limited reasoning power.

Furthermore, we observe a strong locality property of After links by skimming the training data: most After link relations are found in a small local region. Since reasoning and coreference based propagation will accumulate local decisions, a system must be accurate on them.

5.2.1 The Ambiguous Boundary of a Script

Besides the above-mentioned challenges, a more fundamental problem is to define the boundary of scripts. Since the definition of scripts is only prototypical event sequences, the boundaries between them are not clear. In Example 3, the event `jailed` is considered to belong to a "Judicial Process" script and `killing` is considered to belong to an"Attack" script[7]. No link is annotated between these two mentions since they are considered to belong to different clusters, even though the "jailed" event is to punish the "killing". Therefore essentially, the current Event Sequencing task simply requires the system to fit these human defined boundaries. In principle, the "Judicial Process" script and the "Attack" script can form a larger script structure, on a higher hierarchical level.

While it is possible to manually define scripts and what kind of events they may contain specifically in a controlled domain, it is difficult to generalize the relations. Most previous work on script induction (Chambers and Jurafsky, 2008; Cheung et al., 2013; Rudinger et al., 2015; Pichotta and Mooney, 2016; Ferraro and Durme, 2016) treats scripts as statistical models where probabilities can be assigned, thereby avoiding the boundary problem. While the script boundaries may be application dependent, a possible solution may rely on the "Goals" in Schank's script theory. The Goal of a script is the final state expected (by the script protagonist) from the sequence of events. Goal oriented scripts may be able to help us explain whether `killing` and `jailed` should be separate: if we take the"killer" as the protagonist, the goal of "kill" is achieved at the point of the victim dying. We leave the investigation on proper theoretical justification to future work.

6 Conclusion

In this paper, we presented a unified graph framework to conduct event coreference and sequencing. We have achieved state-of-the-art results on event coreference and report the first attempt at event sequencing. While we only studied two types of relations, we believe the method can be adopted in broader contexts. In the future, we plan to build a joint model to allow the tasks to mutually improve each other.

In general, analyzing event structure can bring new aspects of knowledge from text. For instance, Event Coreference systems can help group scattered information together. Understanding Event Sequencing can help clarify the discourse structure, which can be useful in other NLP applications, such as solving entity coreference problems (Peng et al., 2015). However, in our investigation, we find that the linguistic theory and definitions for events are not adequate for the computational setting. For example, proper theoretical justification is needed to define event coreference, which should explain the problems, such as argument mismatches. In addition, we also need a theoretical basis for script boundaries. In the future, we will devote our effort to understanding the theoretical and computational aspects of events relations, and utilizing them for other NLP tasks.

Acknowledgements

This research was supported in part by DARPA grant FA8750-18-2-0018 funded under the AIDA program.

References

Jun Araki, Zhengzhong Liu, Eduard Hovy, and Teruko Mitamura. 2014. Detecting Subevent Structure for Event Coreference Resolution. In Nicoletta Calzolari (Conference Chair) Khalid Choukri Piperidis, Thierry Declerck, Hrafn Loftsson, Bente Maegaard Stelios, Joseph Mariani, Asuncion Moreno, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 4553–4558, Reykjavik, Iceland. European Language Resources Association (ELRA).

Jun Araki. 2018. *Extraction of Event Structures from Text*. Ph.D. thesis, Carnegie Mellon University.

A. Bagga and B. Baldwin. 1998. Entity-based cross-document coreferencing using the vector space model. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics-Volume 1*, pages 79–85. Association for Computational Linguistics.

[7] Script names are taken from the annotation guideline: `http://cairo.lti.cs.cmu.edu/kbp/2016/after/annotation`

Anders Björkelund and Jonas Kuhn. 2014. Learning Structured Perceptrons for Coreference Resolution with Latent Antecedents and Non-local Features. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 47–57.

Nathanael Chambers and Dan Jurafsky. 2008. Unsupervised Learning of Narrative Event Chains. In *ACL '08 Meeting of the Association for Computational Linguistics*, pages 789–797.

Nathanael Chambers and Dan Jurafsky. 2010. A Database of Narrative Schemas. In *LREC 2010, Seventh International Conference on Language Resources and Evaluation*.

Nathanael Chambers and Dan Jurafsky. 2011. Template-Based Information Extraction without the Templates. In *HLT '11 Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, pages 976–986.

Nathanael Chambers, Taylor Cassidy, Bill McDowell, and Steven Bethard. 2014. Dense Event Ordering with a Multi-Pass Architecture. *Transactions of the Association for Computational Linguistics*, 2:273–284.

Zheng Chen and H Ji. 2009. Graph-based event coreference resolution. *Proceedings of the 2009 Workshop on Graph-based Methods for Natural Language Processing*, (August):54–57.

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*, number October, pages 822–828.

Chen Chen and Vincent Ng. 2015. Chinese Event Coreference Resolution : An Unsupervised Probabilistic Model Rivaling Supervised Resolvers. In *Human Language Technologies: The 2015 Annual Conference of the North American Chapter of the ACL*, pages 1097–1107.

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

JC Kit Cheung, H Poon, and Lucy Vanderwende. 2013. Probabilistic Frame Induction. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL/HLT 2013)*.

Nancy Chinchor. 1992. MUC-5 EVALUATION METRIC. In *Proceedings of the 5th Conference on Message Understanding*, pages 69–78.

Michael Collins. 2002. Discriminative Training Methods for Hidden Markov Models: Theory and Experiments with Perceptron Algorithms. In *Proceedings of the Conference on Empirical Methods in NLP (EMNLP 2002)*, number July, pages 1–8.

Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, and Yoram Singer. 2006. Online Passive-Aggressive Algorithms. *Journal of Machine Learning Research*, 7:551–585.

Agata Cybulska and Piek Vossen. 2012. Using Semantic Relations to Solve Event Coreference in Text. In *SemRel2012 in conjunction with LREC2012*, pages 60–67.

Agata Cybulska and Piek Vossen. 2014. Using a sledgehammer to crack a nut? Lexical diversity and event coreference resolution. *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 4545–4552.

Dipanjan Das and NA Smith. 2011. Semi-Supervised Frame-Semantic Parsing for Unknown Predicates. In *HLT '11 Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, volume 1, pages 1435–1444.

Quang Xuan Do, Wei Lu, and Dan Roth. 2012. Joint Inference for Event Timeline Construction. *EMNLP-CoNLL '12 Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, (July):677–687.

Greg Durrett and Dan Klein. 2014. A Joint Model for Entity Analysis: Coreference, Typing, and Linking. *Proceedings of the Transactions of the Association for Computational Linguistics*.

Eraldo Rezende Fernandes, Cícero Nogueira dos Santos, and Ruy Luiz Milidiú. 2012. Latent structure perceptron with feature induction for unrestricted coreference resolution. *Joint Conference on {EMNLP} and {CoNLL-Shared} Task*, pages 41–48.

Francis Ferraro and Benjamin Van Durme. 2016. A Unified Bayesian Model of Scripts , Frames and Language. *Proceedings of the 30th Conference on Artificial Intelligence (AAAI 2016)*, pages 2601–2607.

Laura Hasler and Constantin Orasan. 2009. Do coreferential arguments make event mentions coreferential? In *Proceedings of DAARC*, pages 151–163.

Eduard Hovy, T Mitamura, F Verdejo, J Araki, and Andrew Philpot. 2013. Events are Not Simple: Identity, Non-Identity, and Quasi-Identity. In *The 1st Workshop on EVENTS: Definition, Detection, Coreference and Representation, NAACL-HLT 2013 Workshop*, pages 21–28, Atlanta.

Kevin Humphreys, Robert Gaizauskas, and Saliha Azzam. 1997. Event coreference for information extraction. In *Proceedings of a Workshop on Operational Factors in Practical, Robust Anaphora Resolution for Unrestricted Texts, 35 th Annual Meeting of Assoc. for Computational Linguistics*, pages 75–81, Madrid.

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 Conference on Empirical Methods in Natural Language Processing and Natural Language Learning*.

Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. 2017. End-to-end Neural Coreference Resolution. In *EMNLP 2017*.

Zhengzhong Liu, Jun Araki, Eduard Hovy, and Teruko Mitamura. 2014. Supervised Within-Document Event Coreference using Information Propagation. In Nicoletta Calzolari (Conference Chair) Khalid Choukri Piperidis, Thierry Declerck, Hrafn Loftsson, Bente Maegaard Stelios, Joseph Mariani, Asuncion Moreno, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 4539–4544, Reykjavik, Iceland. European Language Resources Association (ELRA).

Jing Lu and Vincent Ng. 2017. Joint Learning for Event Coreference Resolution. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 90–101.

Jing Lu, Deepak Venugopal, Vibhav Gogate, and Vincent Ng. 2016. Joint Inference for Event Reference Resolution. In *Proceedings of the 26th International Conference on Computational Linguistics*.

Xiaoqiang Luo. 2005. On coreference resolution performance metrics. *HLT '05: Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, (October):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. *Proceedings of 52nd Annual Meeting of the ACL: System Demonstrations*, pages 55–60.

Teruko Mitamura, Zhengzhong Liu, and Eduard Hovy. 2015. Overview of TAC KBP 2015 Event Nugget Track. In *TAC KBP 2015*, pages 1–31.

Teruko Mitamura, Zhengzhong Liu, and Eduard Hovy. 2018. Events Detection, Coreference and Sequencing: What's next? Overview of the TAC KBP 2017 Event Track. In *TAC 2017*, pages 1–42.

Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. 2016. A Corpus and Evaluation Framework for Deeper Understanding of Commonsense Stories. In *Proceedings of NAACL-HLT 2016*, pages 839–849.

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 (ACL)*, number July, pages 104–111, Philadelphia.

Haoruo Peng, Daniel Khashabi, and Dan Roth. 2015. Solving Hard Coreference Problems. In *Human Language Technologies: The 2015 Annual Conference of the North American Chapter of the ACL*, pages 809–819, Denver, Colorado.

Haoruo Peng, Yangqi Song, and Dan Roth. 2016. Event Detection and Co-reference with Minimal Supervision. In *EMNLP 2016*.

Karl Pichotta and Raymond J. Mooney. 2016. Using Sentence-Level LSTM Language Models for Script Inference. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 279–289.

James Pustejovsky, Patrick Hanks, Roser Saurí, Andrew See, Robert Gaizauskas, Andrea Setzer, Beth Sundheim, David Day, Lisa Ferro, and Dragomir. 2002. The TIMEBANK Corpus. *Natural Language Processing and Information Systems*, 4592:647–656.

Marta Recasens and Eduard Hovy. 2011. Blanc: Implementing the rand index for coreference evaluation. *Natural Language Engineering*, 1(1).

Rachel Rudinger, Pushpendre Rastogi, Francis Ferraro, and Benjamin Van Durme. 2015. Script Induction as Language Modeling. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1681–1686.

S Sangeetha and Michael Arock. 2012. Event Coreference Resolution using Mincut based Graph Clustering. *International Journal of Computing & Information Sciences*, pages 253–260.

Roger C Schank and Robert P Abelson. 1977. *Scripts, Plans, Goals and Understanding*. Lawrence Erlbaum Associates.

Stephen Tratz and Eduard Hovy. 2011. A fast, accurate, non-projective, semantically-enriched parser. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, number 2010, pages 1257–1268.

Naushad Uzzaman and James F Allen. 2010. TRIPS and TRIOS System for TempEval-2: Extracting Temporal Information from Text. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, number July, pages 276–283.

Naushad UzZaman, Hector Llorens, Leon Derczynski, Marc Verhagen, James Allen, and James Pustejovsky. 2013. Semeval-2013 task 1: {T}empeval-3: Evaluating time expressions, events, and temporal relations. In *Second joint conference on lexical and computational semantics (* SEM)*, volume 2, pages 1–9.

Naushad UzZaman. 2012. *Interpreting the Temporal Aspects of Language*. Ph.D. thesis, University of Rochester.

Zeno Vendler. 1957. Verbs and times. *The Philosophical Review*, 66(2):143–160.

Piek Vossen and Tommaso Caselli. 2015. Storylines for structuring massive streams of news. In *Proceedings of the First Workshop on Computing News Story Lines*, pages 40–49.