Hierarchical Graph Convolutional Networks for Jointly Resolving Cross-document Coreference of Entity and Event Mentions

Duy Phung[1], **Tuan Ngo Nguyen**[2] **and Thien Huu Nguyen**[2]
[1]VinAI Research, Vietnam
[2]Department of Computer and Information Science,
University of Oregon, Eugene, Oregon, USA
v.duypv1@vinai.io,{tnguyen,thien}@cs.uoregon.edu

Abstract

This paper studies the problem of cross-document event coreference resolution (CDECR) that seeks to determine if event mentions across multiple documents refer to the same real-world events. Prior work has demonstrated the benefits of the predicate-argument information and document context for resolving the coreference of event mentions. However, such information has not been captured effectively in prior work for CDECR. To address these limitations, we propose a novel deep learning model for CDECR that introduces hierarchical graph convolutional neural networks (GCN) to jointly resolve entity and event mentions. As such, sentence-level GCNs enable the encoding of important context words for event mentions and their arguments while the document-level GCN leverages the interaction structures of event mentions and arguments to compute document representations to perform CDECR. Extensive experiments are conducted to demonstrate the effectiveness of the proposed model.

1 Introduction

Event coreference resolution (ECR) aims to cluster event-triggering expressions in text such that all event mentions in a group refer to the same unique event in real world. We are interested in cross-document ECR (CDECR) where event mentions might appear in the same or different documents. For instance, consider the following sentences (event mentions) **S1** and **S2** that involve "*leaving*" and "*left*" (respectively) as event trigger words (i.e., the predicates):

S1: *O'Brien was forced into the drastic step of* **leaving** *the 76ers.*

S2: *Jim O'Brien* **left** *the 76ers after one season as coach.*

An CDECR system in this case would need to recognize that both event mentions in **S1** and **S2** refer to the same event.

A major challenge in CDECR involves the necessity to model entity mentions (e.g., "*Jim O'Brien*") that participate into events and reveal their spatio-temporal information (Yang et al., 2015) (called event arguments). In particular, as event mentions might be presented in different sentences/documents, an important evidence for predicting the coreference of two event mentions is to realize that the two event mentions have the same participants in the real world and/or occur at the same location and time (i.e., same arguments).

Motivated by this intuition, prior work for CDECR has attempted to jointly resolve cross-document coreference for entities and events so the two tasks can mutually benefit from each other (iterative clustering) (Lee et al., 2012). In fact, this iterative and joint modeling approach has recently led to the state-of-the-art performance for CDECR (Barhom et al., 2019; Meged et al., 2020). Our model for CDECR follows this joint coreference resolution method; however, we advance it by introducing novel techniques to address two major limitations from previous work (Yang et al., 2015; Kenyon-Dean et al., 2018; Barhom et al., 2019), i.e., the inadequate mechanisms to capture the argument-related information for representing event mentions and the use of only lexical features to represent input documents.

As the first limitation with the event argument-related evidence, existing methods for CDECR have mainly captured the direct information of event arguments for event mention representations, thus failing to explicitly encode other important context words in the sentences to reveal fine-grained nature of relations between arguments and triggers for ECR (Yang et al., 2015; Barhom et al., 2019). For instance, consider the coreference prediction between the event mentions in **S1** and the following sentence **S3** (with "*leave*" as the event trigger word):

S3: *The baseball coach Jim O'Brien decided to leave the*

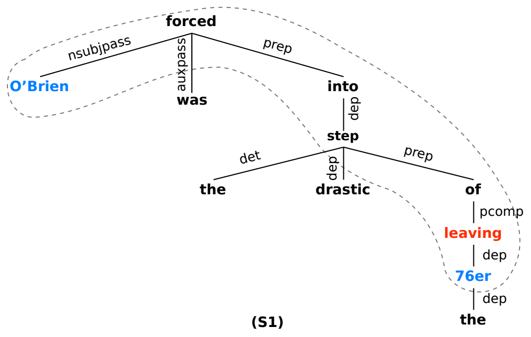

Figure 1: The pruned dependency tree for the event mention in **S1**. The trigger is red while argument heads are blue.

club on Thursday.

The arguments for the event mention in **S3** involves "*The baseball coach Jim O'Brien*", "*the club*", and "*Monday*". If an entity coreference resolution system considers the entity mention pairs ("*O'Brien*" in **S1**, "*The baseball coach Jim O'Brien*" in **S3**) and ("*the 76ers*" in **S1** and "*the club*" in **S3**) as being coreferred, a CDECR system, which only concerns event arguments and their coreference information, would incorrectly predict the coreference between the event mentions in **S1** and **S3** in this case. However, if the CDECR system further models the important words for the relations between event triggers and arguments (i.e., the words "*was forced*" in **S1** and "*decided*" in **S3**), it can realize the unwillingness of the subject for the position ending event in **S1** and the self intent to leave the position for the event in **S3**. As such, this difference can help the system to reject the event coreference for **S1** and **S3**.

To this end, we propose to explicitly identify and capture important context words for event triggers and arguments in representation learning for CDECR. In particular, our motivation is based on the shortest dependency paths between event triggers and arguments that have been used to reveal important context words for their relations (Li et al., 2013; Sha et al., 2018; Veyseh et al., 2020a, 2021). As an example, Figure 1 shows the dependency tree of **S1** where the shortest dependency path between "*O'Brien*" and "*leaving*" can successfully include the important context word "*forced*". As such, for each event mention, we leverage the shortest dependency paths to build a pruned and argument-customized dependency tree to simultaneously contain event triggers, arguments and the important words in a single structure. Afterward, the structure will be exploited to learn richer representation vectors for CDECR.

Second, for document representations, previous work on CDECR has proved that input documents also provide useful context information for event mentions (e.g., document topics) to enhance the clustering performance (Kenyon-Dean et al., 2018). However, the document information is only captured via lexical features in prior work, e.g., TF-IDF vectors (Kenyon-Dean et al., 2018; Barhom et al., 2019), leading to the poor generalization to unseen words/tokens and inability to encapsulate latent semantic information for CDECR. To this end, we propose to learn distributed representation vectors for input documents to enrich event mention representations and improve the generalization of the models for CDECR. In particular, as entity and event mentions are the main objects of interest for CDECR, our motivation is to focus on the context information from these objects to induce document representations for the models. To implement this idea, we propose to represent input documents via interaction graphs between their entity and event mentions, serving as the structures to generate document representation vectors.

Based on those motivations, we introduce a novel hierarchical graph convolutional neural network (GCN) that involves two levels of GCN models to learn representation vectors for the iterative and joint model for CDECR. In particular, sentence-level GCNs will consume the pruned dependency trees to obtain context-enriched representation vectors for event and entity mentions while a document-level GCN will be run over the entity-event interaction graphs, leveraging the mention representations from the sentence-level GCNs as the inputs to generate document representations for CDECR. Extensive experiments show that the proposed model achieves the state-of-the-art resolution performance for both entities and events on the ECB+ dataset. To our knowledge, this is the first work that utilizes GCNs and entity-event interaction graphs for coreference resolution.

2 Related Work

ECR is considered as a more challenging task than entity coreference resolution due to the more complex structures of event mentions that require argument reasoning (Yang et al., 2015). Previous work for within-document event resolution includes pairwise classifiers (Ahn, 2006; Chen et al., 2009), spectral graph clustering methods (Chen and Ji, 2009), information propagation (Liu et al., 2014), markov logic networks (Lu et al., 2016), and deep learning (Nguyen et al., 2016). For only cross-document event resolution, prior work has considered mention-pair classifiers for coreference that use granularities of event slots and lexical features of event mentions for the features (Cybulska and Vossen, 2015b,a). Within- and cross-document event coreference have also been solved simultaneously in previous work (Lee et al., 2012; Bejan and Harabagiu, 2010; Adrian Bejan and Harabagiu, 2014; Yang et al., 2015; Choubey and Huang, 2017; Kenyon-Dean et al., 2018). The most related works to us involve the joint models for entity and event coreference resolution that use contextualized word embeddings to capture the dependencies between the two tasks and lead to the state-of-the-art performance for CDECR (Lee et al., 2012; Barhom et al., 2019; Meged et al., 2020).

Finally, regarding the modeling perspective, our work is related to the models that use GCNs to learn representation vectors for different NLP tasks, e.g., event detection (Lai et al., 2020; Veyseh et al., 2019) and target opinion word extraction (Veyseh et al., 2020b), applying both sentence- and document-level graphs (Sahu et al., 2019; Tran et al., 2020; Nan et al., 2020; Tran and Nguyen, 2021; Nguyen et al., 2021). However, to our knowledge, none of the prior work has employed GCNs for ECR.

3 Model

Given a set of input documents $\mathcal{D}$, the goal of CDECR is to cluster event mentions in the documents of $\mathcal{D}$ according to their coreference. Our model for CDECR follows (Barhom et al., 2019) that simultaneously clusters entity mentions in $\mathcal{D}$ to benefit from the inter-dependencies between entities and events for coreference resolution. In this section, we will first describe the overall framework of our iterative method for joint entity and event coreference resolution based on (Barhom et al., 2019) (a summary of the framework is given in Algorithm 1). The novel hierarchical GCN model for inducing mention[1] representation vectors will be discussed afterward.

Iterative Clustering for CDECR: Following (Barhom et al., 2019), we first cluster the input document set $\mathcal{D}$ into different topics to improve the coreference performance (the set of document topics is called T). As we use the ECB+ dataset (Cybulska and Vossen, 2014) to evaluate the models in this work, the training phase directly utilizes the golden topics of the documents while the test phase applies the K-mean algorithm for document clustering as in (Barhom et al., 2019). Afterward, given a topic $t \in T$ with the corresponding document subset $\mathcal{D}_t \subset \mathcal{D}$, our CDECR model initializes the entity and event cluster configurations E_t^0 and V_t^0 (respectively) where: E_t^0 involves within-document clusters of the entity mentions in the documents in $\mathcal{D}_t$, and V_t^0 simply puts each event mention presented in $\mathcal{D}_t$ into its own cluster (lines 2 and 3 in Algorithm 1). In the training phase, E_t^0 is obtained from the golden within-document coreference information of the entity mentions (to reduce noise) while the within-document entity mention clusters returned by Stanford CoreNLP (Manning et al., 2014) are used for E_t^0 in the test phase, following (Barhom et al., 2019). For convenience, the sets of entity and event mentions in $\mathcal{D}_t$ are called M_t^E and M_t^V respectively.

Algorithm 1 Training algorithm

```
1: for $t \in T$ do
2:     $E_t^0 \leftarrow$ Within-doc clusters of entity mentions
3:     $V_t^0 \leftarrow$ Singleton event mentions in $M_t^V$
4:     $k \leftarrow 1$
5:     while $\exists$ meaningful cluster-pair merge do
6:         //Entities
7:         Generate entity mention representations $R_E(m_{e_i}, V_t^{k-1})$ for all $m_{e_i} \in M_t^E$
8:         Compute entity mention-pair coreference scores $S_E(m_{e_i}, m_{e_j})$
9:         Train $R_E$ and $S_E$ using the gold entity mention clusters
10:        $E_t^k \leftarrow$ Agglomeratively cluster $M_t^E$ based on $S_E(m_{e_i}, m_{e_j})$
11:        //Events
12:        Generate event mention representations $R_V(m_{v_i}, E_t^k)$ for all $m_{v_i} \in M_t^V$
13:        Compute event mention-pair coreference scores $S_V(m_{v_i}, m_{v_j})$
14:        Train $R_V$ and $S_V$ using the gold entity mention clusters
15:        $V_t^k \leftarrow$ Agglomeratively cluster $M_t^V$ based on $S_V(m_{v_i}, m_{v_j})$
16:        $k \leftarrow k + 1$
17:    end while
18: end for
```

Given the initial configurations, our iterative algorithm involves a sequence of clustering iterations, generating new cluster configurations E_t^k and V_t^k for entities and events (respectively) after each iteration. As such, each iteration k performs two independent clustering steps where entity mentions are clustered first to produce E_t^k, followed by event mention clustering to obtain

[1]We use mentions to refer to both event and entity mentions.

V_t^k (alternating the clustering). Starting with the entity clustering at iteration k, each entity mention m_{e_i} is first transformed into a representation vector $R_E(m_{e_i}, V_t^{k-1})$ (line 7 in Algorithm 1) that is conditioned on not only the specific context of m_{e_i} but also the current event cluster configuration V_t^{k-1} (i.e., to capture the event-entity inter-dependencies). Afterward, a scoring function $S_E(m_{e_i}, m_{e_j})$ is used to compute the coreference probability/score for each pair of entity mentions, leveraging their mention representations $R_E(m_{e_i}, V_t^{k-1})$ and $R_E(m_{e_j}, V_t^{k-1})$ at the current iteration (R_E and S_E will be discussed in the next section) (line 8). An agglomerative clustering algorithm then utilizes these coreference scores to cluster the entity mentions, leading to a new configuration E_t^k for entity mention clusters (line 10). Given E_t^k, the same procedure is applied to cluster the event mentions for iteration k, including: (i) obtaining an event representation vector $R_V(m_{v_i}, E_t^k)$ for each event mention m_{v_i} based on the current entity cluster configuration E_t^k, (ii) computing coreference scores for the event mention pairs via the scoring function $S_V(m_{v_i}, m_{v_j})$, and (iii) performing agglomerative clustering for the event mentions to produce the new configuration V_t^k for event clusters (lines 12, 13, and 15).

Note that during the training process, the parameters of the representation and scoring functions for entities R_E and S_E (or for events with R_V and S_V) are updated/optimized after the coreference scores are computed for all the entity (or event) mention pairs. This corresponds to lines 9 and 14 in Algorithm 1 (not used in the test phase). In particular, the loss function to optimize R_E and S_E in line 9 is based on the cross-entropy over every pair of entity mentions (m_{e_i}, m_{e_j}) in M_t^E: $L_t^{ent,coref} = -\sum_{m_{e_i} \neq m_{e_j}} y_{ij}^{ent} \log S_E(m_{e_i}, m_{e_j}) - (1 - y_{ij}^{ent}) \log(1 - S_E(m_{e_i}, m_{e_j})$ where y_{ij}^{ent} is a golden binary variable to indicate whether m_{e_i} and m_{e_j} corefer or not (symmetrically for $L_t^{env,coref}$ to train R_V and S_V in line 14). Also, we use the predicted configurations E_t^k and V_t^{k-1} in the mention representation computation (instead of the golden clusters as in y_{ij}^{ent} for the loss functions) during both the training and test phases to achieve a consistency (Barhom et al., 2019).

Finally, we note that the agglomerative clustering in each iteration of our model also starts with the initial clusters as in E_t^0 and V_t^0, and greedily merges multiple cluster pairs with the highest cluster-pair scores until all the scores are below a predefined threshold δ. In this way, the algorithm first focuses on high-precision merging operations and postpones less precise ones until more information is available. The cluster-pair score $S_C(c_i, c_j)$ for two clusters c_i and c_j (mention sets) at some algorithm step is based on averaging mention linkage coreference scores: $S_C(c_i, c_j) = \frac{1}{|c_i||c_j|} \sum_{m_i \in c_i, m_j \in c_j} S_*(m_i, m_j)$ (Barhom et al., 2019) where $*$ can be E or V depending on whether c_i and c_j are entity or event clusters (respectively).

Mention Representations: Let m be a mention (event or entity) in a sentence $W = w_1, w_2, \ldots, w_n$ of n words (w_i is the i-th word) where w_a is the head word of m. To prepare W for the mention representation computation and achieve a fair comparison with (Barhom et al., 2019), we first convert each word $w_i \in W$ into a vector x_i using the ELMo embeddings (Peters et al., 2018). Here, x_i is obtained by running the pre-trained ELMo model over W and averaging the hidden vectors for w_i at the three layers in ELMo. This transforms W into a sequence of vectors $X = x_1, x_2, \ldots, x_n$ for the next steps. The mention representations in our work are based on two major elements, i.e., the modeling of important context words for event triggers and arguments, and the induction of document presentations.

(i) Modeling Important Context Words: A motivation for representation learning in our model is to capture event arguments and important context words (for the relations between event triggers and arguments) to enrich the event mention representations. In this work, we employ a symmetric intuition to compute a representation vector for an entity mention m, aiming to encode associated predicates (i.e., event triggers that accept m as an argument W) and important context words (for the relations between the entity mention and associated event triggers). As such, following (Barhom et al., 2019), we first identify the attached arguments (if m is an event mention) or predicates (if m is an entity mention) in W for m using a semantic role labeling (SRL) system. In particular, we focus on four semantic roles of interest: `Arg0`, `Arg1`, `Location`, and `Time`. For convenience, let $A^m = \{w_{i_1}, \ldots, w_{i_o}\} \subset W$ be the set of head words of the attached event arguments or event triggers for m in W based on the SRL system and the four roles (o is the number of head words). In particular, if m is an event mention,

A^m would involve the head words of the entity mentions that fill the four semantic roles for m in W. In contrast, if m is an entity mention, A^m would capture the head words of the event triggers that take m as an argument with one of the four roles in W. Afterward, to encode the important context words for the relations between m and the words in A^m, we employ the shortest dependency paths P_{i_j} between the head word w_a of m and the words $w_{i_j} \in A^m$. As such, starting with the dependency tree of $\mathcal{G}^m = \{\mathcal{N}^m, \mathcal{E}^m\}$ of W ($\mathcal{N}^m$ involves the words in W), we build a pruned tree $\hat{\mathcal{G}}^m = \{\hat{\mathcal{N}}^m, \hat{\mathcal{E}}^m\}$ of $\mathcal{G}^m$ to explicitly focus on the mention m and its important context words in the paths P_{i_j}. Concretely, the node set $\hat{\mathcal{N}}^m$ of $\hat{\mathcal{G}}^m$ contains all the words in the paths P_{i_j} ($\hat{\mathcal{N}}^m = \bigcup_{j=1..o} P_{i_j}$) while the edge set $\hat{\mathcal{E}}^m$ preserves the dependency connections in $\mathcal{G}^m$ of the words in $\hat{\mathcal{N}}^m$ ($\hat{\mathcal{E}}^m = \{(a,b) | a, b \in \hat{\mathcal{N}}^m, (a,b) \in \mathcal{E}^m\}$). As such, the pruned tree $\hat{\mathcal{G}}^m$ helps to gather the relevant context words for the coreference resolution of m and organize them into a single dependency graph, serving as a rich structure to learn a representation vector for m[2] (e.g., in Figure 1).

(ii) Graph Convolutional Networks: The context words and structure in $\hat{\mathcal{G}}^m$ suggest the use of Graph Convolutional Networks (GCN) (Kipf and Welling, 2017; Nguyen and Grishman, 2018) to learn the representation vector for m (called the sentence-level GCN). In particular, the GCN model in this work involves several layers (i.e., L layers in our case) to compute the representation vectors for the nodes in the graph $\hat{\mathcal{G}}^m$ at different abstract levels. The input vector h_v^0 for the node $v \in \hat{\mathcal{N}}^m$ for GCN is set to the corresponding ELMo-based vectors in X. After L layers, we obtain the hidden vectors h_v^L (in the last layer) for the nodes $v \in \hat{\mathcal{N}}^m$. We call $sent(m) = [h_{v_m}^L, max\_pool(h_v^L | v \in \hat{\mathcal{N}}^m)]$ the sentence-level GCN vector that will be used later to represent m (v_m is the corresponding node of the head word of m in $\hat{\mathcal{N}}^m$).

(iii) GCN Interaction: Our discussion about the sentence-level GCN so far has been agnostic to whether m is an event or entity mention and the straightforward approach is to apply the same GCN model for both entity and event mentions. However, this approach might limit the flexibility of the GCN model to focus on the necessary aspects of information that are specific to each coreference resolution task (i.e., events or entities). For example, event coreference might need to weight the information from event arguments and important context words more than those for entity coreference (Yang et al., 2015). To this end, we propose to apply two separate sentence-level GCN models for event and entity mentions, which share the architecture but differ from the parameters, to enhance representation learning (called G^{ent} and G^{evn} for entities and events respectively). In addition, to introduce a new source of training signals and promote the knowledge transfer between the two GCN networks, we propose to regularize the GCN models so they produce similar representation vectors for the same input sentence W. In particular, we apply both GCN models G^{ent} and G^{evn} over the full dependency tree[3] $\mathcal{G}^m$ using the ELMo-based vectors X for W as the input. This produces the hidden vectors $h_1^{ent}, \ldots, h_n^{ent}$ and $h_1^{env}, \ldots, h_n^{env}$ in the last layers of G^{ent} and G^{evn} for W. Afterward, we compute two versions of representation vectors for W based on max-pooling: $h^{ent} = max\_pool(h_1^{ent}, \ldots, h_n^{ent})$, $h^{env} = max\_pool(h_1^{env}, \ldots, h_n^{env})$. Finally, the mean squared difference between h^{ent} and h^{env} is introduced into the overall loss function to regularize the models: $L_m^{reg} = ||h^{ent} - h^{env}||_2^2$. As this loss is specific to mention m, it will be computed independently for event and entity mentions and added into the corresponding training loss (i.e., lines 9 or 14 in Algorithm 1). In particular, the overall training loss for entity mention coreference resolution in line 9 of Algorithm 1 is: $L_t^{ent} = \alpha^{ent} L_t^{ent,coref} + (1 - \alpha^{ent}) \sum_{m_e \in M_t^E} L_{m_e}^{reg}$ while those for event mention coreference resolution is: $L_t^{env} = \alpha^{env} L_t^{env,coref} + (1 - \alpha^{env}) \sum_{m_v \in M_t^V} L_{m_v}^{reg}$ (line 14). Here, α^{ent} and α^{env} are the trade-off parameters.

(iv) Document Representation: As motivated in the introduction, we propose to learn representation vectors for input documents to enrich mention representations and improve the generalization of the models (over the lexical features for documents). As such, our principle is to employ entity and event mentions (the main objects of interest in CDECR) and their interactions/structures to represent input documents (i.e., documents as interaction graphs of entity and event mentions). Given the mention m of interest and its correspond-

[2]If A^m is empty, the pruned tree $\hat{\mathcal{G}}^m$ only contains the head word of m.

[3]We tried the pruned dependency tree $\hat{\mathcal{G}}^m$ in this regularization, but the full dependency tree $\mathcal{G}^m$ led to better results.

ing document $d \in \mathcal{D}_t$, we start by building an interaction graph $\mathcal{G}^{doc} = \{\mathcal{N}^{doc}, \mathcal{E}^{doc}\}$ where the node set $\mathcal{N}^{doc}$ involves all the entity and event mentions in d. For the edge set $\mathcal{E}^{doc}$, we leverage two types of information to connect the nodes in $\mathcal{N}^{doc}$: (i) predicate-argument information: we establish a link between the nodes for an event mention x and an entity mention y in $\mathcal{N}^{doc}$ if y is an argument of x for one of the four semantic roles, i.e., `Arg0`, `Arg1`, `Location`, and `Time` (identified by the SRL system), and (ii) entity mention coreference: we connect any pairs of nodes in $\mathcal{N}^{doc}$ that correspond to two coreferring entity mentions in d (using the gold within-document entity coreference in training and the predicted one in testing, as in E_t^0). In this way, $\mathcal{G}^{doc}$ helps to emphasize on important objects of d and enables intra- and inter-sentence interactions between event mentions (via entity mentions/arguments) to produce effective document representations for CDECR.

In the next step, we feed the interaction graph $\mathcal{G}^{doc}$ into a GCN model G^{doc} (called the document-level GCN) using the sentence-level GCN-based representations of the entity and event mentions in $\mathcal{N}^{doc}$ (i.e., $sent(m)$) as the initial vectors for the nodes (thus called a hierarchical GCN model). The hidden vectors produced by the last layer of G^{doc} for the nodes in $\mathcal{G}^{doc}$ is called $\{h_u^d | u \in \mathcal{N}^{doc}\}$. Finally, we obtain the representation vector $doc(m)$ for d based on the max-pooling: $doc(m) = max\_pool(h_u^d | u \in \mathcal{N}^{doc})$.

(v) Final Representation: Given the representation vectors learned so far, we form the final representation vector for m ($R_E(m, V_t^{k-1})$ or $R_V(m, E_t^k)$ in lines 7 or 12 of Algorithm 1) by concatenating the following vectors:

(1) The sentence- and document-level GCN-based representation vectors for m (i.e., $sent(m)$ and $doc(m)$).

(2) The cluster-based representation $cluster(m) = [\texttt{Arg0}_m, \texttt{Arg1}_m, \texttt{Location}_m, \texttt{Time}_m]$. Taking $\texttt{Arg0}_m$ as an example, it is computed by considering the mention m' that is associated with m via the semantic role `Arg0` in W using the SRL system. Here, m' is an event mention if m is an entity mention and vice versa (`Arg0` = 0 if m' does not exist). As such, let c be the cluster in the current configuration (i.e., V_t^{k-1} or E_t^k) that contain m'. We then obtain $\texttt{Arg0}_m$ by averaging the ELMo-based vectors (i.e., $X = x_1, \ldots, x_n$) of the head words of the mentions in c: $\texttt{Arg0}_m = 1/|c| \sum_{q \in c} x_{head(q)}$ ($head(q)$ is the index of the head word of mention q in W). Note that as the current entity cluster configuration E_t^k is used to generate the cluster-based representations if m is an event mention (and vice verse), it serves as the main mechanism to enable the two coreference tasks to interact and benefit from each other. These vectors are inherited from (Barhom et al., 2019) for a fair comparison.

Finally, given two mentions m_1 and m_2, and their corresponding representation vectors $R(m_1)$ and $R(m_2)$ (as computed above), the coreference score functions S_E and S_V send the concatenated vector $[R(m_1), R(m_2), R(m_1) \odot R(m_2)]$ to two-layer feed-forward networks (separate ones for S_E and S_V) that involve the sigmoid function in the end to produce coreference score for m_1 and m_2. Here, $\odot$ is the element-wise product. This completes the description of our CDECR model.

4 Experiments

Dataset: We use the ECB+ dataset (Cybulska and Vossen, 2014) to evaluate the CDECR models in this work. Note that ECB+ is the largest dataset with both within- and cross-document annotation for the coreference of entity and event mentions so far. We follow the setup and split for this dataset in prior work to ensure a fair comparison (Cybulska and Vossen, 2014; Kenyon-Dean et al., 2018; Barhom et al., 2019; Meged et al., 2020). In particular, this setup employs the annotation subset that has been validated for correctness by (Cybulska and Vossen, 2014) and involves a larger portion of the dataset for training. In ECB+, only a part of the mentions are annotated. This setup thus utilizes gold-standard event and entity mentions in the evaluation and does not require special treatment for unannotated mentions (Barhom et al., 2019).

Note that there is a different setup for ECB+ that is applied in (Yang et al., 2015) and (Choubey and Huang, 2017). In this setup, the full ECB+ dataset is employed, including the portions with known annotation errors. In test time, such prior work utilizes the predicted mentions from a mention extraction tool (Yang et al., 2015). To handle the partial annotation in ECB+, those prior work only evaluates the systems on the predicted mentions that are also annotated as the gold mentions. However, as shown by (Upadhyay et al., 2016), this ECB+ setup has several limitations (e.g., the ignorance of clusters with a single mention and the separate evaluation for each sub-topic). Following (Barhom et al., 2019), we thus do not evaluate the systems on this setup, i.e., not comparing our model with those models in (Yang et al., 2015) and (Choubey and Huang, 2017) due to the incompatibility.

Hyper-Parameters: To achieve a fair comparison, we utilize the preprocessed data and extend the implementation for the model in (Barhom et al., 2019) to include our novel hierarchical GCN model. The development dataset of ECB+ is used to tune the hyper-parameters of the proposed model (called HGCN). The suggested values and the resources for our model are reported in Appendix A.

Comparison: Following (Barhom et al., 2019), we compare HGCN with the following baselines:

(i) **LEMMA** (Barhom et al., 2019): This first clusters documents to topics and then groups event mentions that are in the same document clusters and share the head lemmas.

(ii) **CV** (Cybulska and Vossen, 2015b): This is a supervised learning method for CDECR that leverages discrete features to represent event mentions and documents. We compare with the best reported results for this method as in (Barhom et al., 2019).

(iii) **KCP** (Kenyon-Dean et al., 2018): This is a neural network model for CDECR. Both event mentions and document are represented via word embeddings and hand-crafted binary features.

(iv) **C-KCP** (Barhom et al., 2019): This is the KCP model that is retrained and tuned using the same document clusters in the test phase as in (Barhom et al., 2019) and our model.

(v) **BSE** (Barhom et al., 2019): This is a joint resolution model for cross-document coreference of entity and event mentions, using ELMo to compute representations for the mentions.

(v) **BSE-DJ** (Barhom et al., 2019): This is a variant of BSE that does not use the cluster-based representations $cluster(m)$ in the mention representations, thus performing event and entity coreference resolution separately.

(vii) **MCS** (Meged et al., 2020): An extension of BSE where some re-ranking features are included. Note that BSE and MCS are the current state-of-the-art (SOTA) models for CDECR on ECB+.

For cross-document entity coreference resolution, we compare our model with the LEMMA and BSE models in (Barhom et al., 2019), the only works that report the performance for event mentions on ECB+ so far. Following (Barhom et al., 2019), we use the common coreference resolution metrics to evaluate the models in this work, including MUC (Vilain et al., 1995), B^3, CEAF-e (Luo, 2005), and CoNLL F1 (average of three previous metrics). The official CoNLL scorer in (Pradhan et al., 2014) is employed to compute these metrics. Tables 1 and 2 show the performance (F1 scores) of the models for cross-document resolution for entity and event mentions (respectively). Note that we also report the performance of a variant (called **HGCN-DJ**) of the proposed HGCN model where the cluster-based representations $cluster(m)$ are excluded (thus separately doing event and entity resolution as BSE-DJ).

Model	**MUC**	**B^3**	**CEAF-e**	**CoNLL**
LEMMA (Barhom et al., 2019)	78.1	77.8	73.6	76.5
CV (Cybulska and Vossen, 2015b)	73.0	74.0	64.0	73.0
KCP (Kenyon-Dean et al. 2019)	69.0	69.0	69.0	69.0
CKCP (Barhom et al., 2019)	73.4	75.9	71.5	73.6
BSE-DJ (Barhom et al., 2019)	79.4	80.4	75.9	78.5
BSE (Barhom et al., 2019)	80.9	80.3	77.3	79.5
MCS (Meged et al., 2020)	81.6	80.5	77.8	80.0
HGCN-DJ	81.3	80.8	76.1	79.4
HGCN (proposed)	**83.1**	**82.1**	**78.8**	**81.3**

Table 1: The cross-document event coreference resolution performance (F1) on the ECB+ test set.

Model	**MUC**	**B^3**	**CEAF-e**	**CoNLL**
LEMMA (Barhom et al., 2019)	76.7	65.6	60.0	67.4
BSE-DJ (Barhom et al., 2019)	78.7	69.9	61.6	70.0
BSE (Barhom et al., 2019)	79.7	70.5	63.3	71.2
HGCN-DJ	80.1	70.7	62.2	71.0
HGCN (proposed)	**82.1**	**71.7**	**63.4**	**72.4**

Table 2: The cross-document entity coreference resolution performance (F1) on the ECB+ test set.

As can be seen, HGCN outperforms the all the baselines models on both entity and event coreference resolution (over different evaluation metrics). In particular, the CoNLL F1 score of HGCN for event coreference is 1.3% better than those for MCS (the prior SOTA model) while the CoNLL F1 improvement of HGCN over BSE (the prior SOTA model for entity coreference on ECB+) is 1.2%. These performance gaps are significant with $p < 0.001$, thus demonstrating the effectiveness of the proposed model for CDECR. Importantly, HGCN is significantly better than BSE, the most direct baseline of the proposed model, on both entity and event coreference regardless of whether the cluster-based representations $cluster(m)$ for joint entity and event resolution is used or not. This testifies to the benefits of the proposed hierarchical model for representation learning for CDECR. Finally, we evaluate the full HGCN model when ELMo embeddings are replaced with BERT embeddings (Devlin et al., 2019), leading to the CoNLL F1 scores of 79.7% and 72.3% for event and entity coreference (respectively). This performance is either worse (for events) or comparable (for entities) than those for EMLo, thus showing the advantages of ELMo for our tasks on ECB+.

Ablation Study: To demonstrate the benefits of the proposed components for our CDECR model, we evaluate three groups of ablated/varied models for HGCN. First, for the effectiveness of the pruned dependency tree $\hat{\mathcal{G}}^m$ and the sentence-level GCN models G^{ent} and G^{env}, we consider the following baselines: (i) **HGCN-Sentence GCNs**: this model removes the sentence-level GCN models G^{ent} and G^{env} from HGCN and directly feed the ELMo-based vectors X of the mention heads into the document-level GCN G^{doc} (the representation vector $sent(m)$ is thus not included), and (ii) **HGCN-Pruned Tree**: this model replaces the pruned dependency tree $\hat{\mathcal{G}}^m$ with the full dependency tree $\mathcal{G}^m$ in the computation. Second, for the advantage of the GCN interaction between G^{ent} and G^{env}, we examine two baselines: (iii) **HGCN with One Sent GCN**: this baseline only uses one sentence-level GCN model for both entity and event mentions (the regularization loss L^{reg}_m for G^{ent} and G^{env} is thus not used as well), and (iv) **HGCN-L^{reg}_m**: this baseline still uses two sentence-level GCN models but excludes the regularization term L^{reg}_m from the training losses. Finally, for the benefits of the document-level GCN G^{doc}, we study the following variants: (v) **HGCN-G^{doc}**: this model removes the document-level GCN model from HGCN, thus excluding the document representations $doc(m)$ from the mention representations, (vi) **HGCN-G^{doc}+TFIDF**: this model also excludes G^{doc}, but it includes the TF-IDF vectors for documents, based on uni-, bi- and tri-grams, in the mention representations (inherited from (Kenyon-Dean et al., 2018)), and (vii) **HGCN-G^{doc}+MP**: instead of using the GCN model G^{doc}, this model aggregates mention representation vectors produced by the sentence-level GCNs ($sent(m)$) to obtain the document representations $doc(m)$ using max-pooling. Table 3 presents the performance of the models for event coreference on the ECB+ test set.

Model	MUC	B^3	CEAF-e	CoNLL
HGCN (full)	**83.1**	**82.1**	**78.8**	**81.3**
HGCN-Sentence GCNs	79.7	80.7	76.4	79.0
HGCN-Pruned Tree	81.8	81.1	77.1	80.0
HGCN with One Sent GCN	82.1	81.0	76.3	79.8
HGCN-L^{reg}_m	81.6	81.6	77.6	80.3
HGCN-G^{doc}	82.6	81.8	76.7	80.4
HGCN-G^{doc}+TFIDF	82.2	81.0	78.4	80.5
HGCN-G^{doc}+MP	82.0	81.1	77.0	80.0

Table 3: The CDECR F1 scores on the ECB+ test set.

It is clear from the table that all the ablated baselines are significantly worse than the full model HGCN (with $p < 0.001$), thereby confirming the necessity of the proposed GCN models (the sentence-level GCNs with pruned trees and document-level GCN) and the GCN interaction mechanism for HGCN and CDECR. In addition, the same trends for the model performance are also observed for entity coreference in this ablation study (the results are shown in Appendix B), thus further demonstrating the benefits of the proposed components in this work. Finally, we show the distribution of the error types of our HGCN model in Appendix C for future improvement.

5 Conclusion

We present a model to jointly resolve the cross-document coreference of entity and event mentions. Our model introduces a novel hierarchical GCN that captures both sentence and document context for the representations of entity and event mentions. In particular, we design pruned dependency trees to capture important context words for sentence-level GCNs while interaction graphs between entity and event mentions are employed for document-level GCN. In the future, we plan to explore better mechanisms to identify important context words for CDECR.

Acknowledgments

This research has been supported by the Army Research Office (ARO) grant W911NF-21-1-0112. This research is also based upon work supported by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via IARPA Contract No. 2019-19051600006 under the Better Extraction from Text Towards Enhanced Retrieval (BETTER) Program. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of ARO, ODNI, IARPA, the Department of Defense, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein. This document does not contain technology or technical data controlled under either the U.S. International Traffic in Arms Regulations or the U.S. Export Administration Regulations.

References

Cosmin Adrian Bejan and Sanda Harabagiu. 2014. Unsupervised event coreference resolution. In *Computational Linguistics*.

David Ahn. 2006. The stages of event extraction. In *Proceedings of the Workshop on Annotating and Reasoning about Time and Events*.

Shany Barhom, Vered Shwartz, Alon Eirew, Michael Bugert, Nils Reimers, and Ido Dagan. 2019. Revisiting joint modeling of cross-document entity and event coreference resolution. In *ACL*.

Cosmin Bejan and Sanda Harabagiu. 2010. Unsupervised event coreference resolution with rich linguistic features. In *ACL*.

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

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 Workshop on Events in Emerging Text Types*.

Prafulla Kumar Choubey and Ruihong Huang. 2017. Event coreference resolution by iteratively unfolding inter-dependencies among events. In *EMNLP*.

Agata Cybulska and Piek Vossen. 2014. Using a sledgehammer to crack a nut? lexical diversity and event coreference resolution. In *LREC*.

Agata Cybulska and Piek Vossen. 2015a. Translating granularity of event slots into features for event coreference resolution. In *Proceedings of the The 3rd Workshop on EVENTS: Definition, Detection, Coreference, and Representation*.

Agata Cybulska and Piek T. J. M. Vossen. 2015b. "bag of events" approach to event coreference resolution. supervised classification of event templates. In *Int. J. Comput. Linguistics Appl.*

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*.

Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. In *To appear*.

Kian Kenyon-Dean, Jackie Chi Kit Cheung, and Doina Precup. 2018. Resolving event coreference with supervised representation learning and clustering-oriented regularization. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *ICLR*.

Viet Dac Lai, Tuan Ngo Nguyen, and Thien Huu Nguyen. 2020. Event detection: Gate diversity and syntactic importance scores for graph convolution neural networks. In *EMNLP*.

Heeyoung Lee, Marta Recasens, Angel Chang, Mihai Surdeanu, and Dan Jurafsky. 2012. Joint entity and event coreference resolution across documents. In *EMNLP*.

Qi Li, Heng Ji, and Liang Huang. 2013. Joint event extraction via structured prediction with global features. In *ACL*.

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

Jing Lu, Deepak Venugopal, Vibhav Gogate, and Vincent Ng. 2016. Joint inference for event coreference resolution. In *COLING*.

Xiaoqiang Luo. 2005. On coreference resolution performance metrics. In *EMNLP*.

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

Yehudit Meged, Avi Caciularu, Vered Shwartz, and Ido Dagan. 2020. Paraphrasing vs coreferring: Two sides of the same coin. In *ArXiv abs/2004.14979*.

Guoshun Nan, Zhijiang Guo, Ivan Sekulic, and Wei Lu. 2020. Reasoning with latent structure refinement for document-level relation extraction. In *ACL*.

Minh Van Nguyen, Viet Dac Lai, and Thien Huu Nguyen. 2021. Cross-task instance representation interactions and label dependencies for joint information extraction with graph convolutional networks. In *NAACL-HLT*.

Thien Huu Nguyen, , Adam Meyers, and Ralph Grishman. 2016. New york university 2016 system for kbp event nugget: A deep learning approach. In *TAC*.

Thien Huu Nguyen and Ralph Grishman. 2018. Graph convolutional networks with argument-aware pooling for event detection. In *AAAI*.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. In *Journal of Machine Learning Research*.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *NAACL-HLT*.

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

Sunil Kumar Sahu, Fenia Christopoulou, Makoto Miwa, and Sophia Ananiadou. 2019. Inter-sentence relation extraction with document-level graph convolutional neural network. In *ACL*.

Lei Sha, Feng Qian, Baobao Chang, and Zhifang Sui. 2018. Jointly extracting event triggers and arguments by dependency-bridge rnn and tensor-based argument interaction. In *AAAI*.

M. Surdeanu, Lluís Màrquez i Villodre, X. Carreras, and P. Comas. 2007. Combination strategies for semantic role labeling. In *Journal of Artificial Intelligence Research*.

Hieu Minh Tran, Minh Trung Nguyen, and Thien Huu Nguyen. 2020. The dots have their values: Exploiting the node-edge connections in graph-based neural models for document-level relation extraction. In *EMNLP Findings*.

Minh Tran and Thien Huu Nguyen. 2021. Graph convolutional networks for event causality identification with rich document-level structures. In *NAACL-HLT*.

Shyam Upadhyay, Nitish Gupta, Christos Christodoulopoulos, and Dan Roth. 2016. Revisiting the evaluation for cross document event coreference. In *COLING*.

Amir Pouran Ben Veyseh, Franck Dernoncourt, Quan Hung Tran, Varun Manjunatha, Lidan Wang, Rajiv Jain, Doo Soon Kim, Walter Chang, and Thien Huu Nguyen. 2021. Inducing rich interaction structures between words for document-level event argument extraction. In *Proceedings of the 25th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*.

Amir Pouran Ben Veyseh, Thien Huu Nguyen, and Dejing Dou. 2019. Graph based neural networks for event factuality prediction using syntactic and semantic structures. In *ACL*.

Amir Pouran Ben Veyseh, Tuan Ngo Nguyen, and Thien Huu Nguyen. 2020a. Graph transformer networks with syntactic and semantic structures for event argument extraction. In *EMNLP Findings*.

Amir Pouran Ben Veyseh, Nasim Nouri, Franck Dernoncourt, Dejing Dou, and Thien Huu Nguyen. 2020b. Introducing syntactic structures into target opinion word extraction with deep learning. In *EMNLP*.

Marc Vilain, John Burger, John Aberdeen, Dennis Connolly, and Lynette Hirschman. 1995. A model-theoretic coreference scoring scheme. In *Sixth Message Understanding Conference (MUC-6)*.

Bishan Yang, Claire Cardie, and Peter Frazier. 2015. A hierarchical distance-dependent Bayesian model for event coreference resolution. In *TACL*.