Learning Cross-Task Dependencies for Joint Extraction of Entities, Events, Event Arguments, and Relations

Minh Van Nguyen[1], **Bonan Min**[2*], **Franck Dernoncourt**[3], **and Thien Huu Nguyen**[1,4]
[1] Dept. of Computer Science, University of Oregon, Eugene, OR, USA
[2] Amazon AWS AI Labs
[3] Adobe Research, Seattle, WA, USA
[4] VinAI Research, Vietnam
{minhnv,thien}@cs.uoregon.edu,
bonanmin@amazon.com, dernonco@adobe.com

Abstract

Extracting entities, events, event arguments, and relations (i.e., task instances) from text represents four main challenging tasks in information extraction (IE), which have been solved jointly (JointIE) to boost the overall performance for IE. As such, previous work often leverages two types of dependencies between the tasks, i.e., cross-instance and cross-type dependencies representing relatedness between task instances and correlations between information types of the tasks. However, the cross-task dependencies in prior work are not optimal as they are only designed manually according to some task heuristics. To address this issue, we propose a novel model for JointIE that aims to learn cross-task dependencies from data. In particular, we treat each task instance as a node in a dependency graph where edges between the instances are inferred through information from different layers of a pretrained language model (e.g., BERT). Furthermore, we utilize the Chow-Liu algorithm to learn a dependency tree between information types for JointIE by seeking to approximate the joint distribution of the types from data. Finally, the Chow-Liu dependency tree is used to generate cross-type patterns, serving as anchor knowledge to guide the learning of representations and dependencies between instances for JointIE. Experimental results show that our proposed model significantly outperforms strong JointIE baselines over four datasets with different languages.

1 Introduction

Entity mention recognition (EMR), event trigger detection (ETD), event argument extraction (EAE), and relation extraction (RE) are four main challenging tasks in information extraction (IE), which aim to extract entities (e.g., a person), events (e.g., an attack), event arguments (e.g., a victim in an attack), and relations (e.g., work-for) mentioned in text. These IE tasks have been solved mostly in pipelined approaches (Li et al., 2013; Nguyen and Grishman, 2015; Chen et al., 2015; Lai et al., 2020; Du and Cardie, 2020; Veyseh et al., 2020; Li et al., 2020; Pouran Ben Veyseh et al., 2021; Nguyen et al., 2021b), where input to a model performing an IE task involves predictions from other models performing other IE tasks. As a result, errors in predictions by a model can be propagated to subsequent models in the pipeline to hurt overall performance.

To avoid error propagation, the four IE tasks can be solved jointly (JointIE) in a single model (Lin et al., 2020; Nguyen et al., 2021a; Zhang and Ji, 2021). As such, a key challenge for JointIE models is to effectively capture dependencies between the IE tasks to boost overall extraction performance. In particular, two types of task dependencies are important for JointIE, i.e., cross-instance and cross-type dependencies. First, for cross-instance dependencies, JointIE models use instances to refer to word spans for event triggers/entity mentions (for EMR and ETD) or pair of word spans of event triggers/entity mentions (for EAE and RE) that should be classified according to predefined information types for IE. Accordingly, an important insight from previous JointIE models is to enrich the representation for one instance with those from related instances in different IE tasks to facilitate the type prediction (Lin et al., 2020; Nguyen et al., 2021a). To this end, a typical approach to encode cross-instance dependencies for representation learning in previous work involves creating dependency graphs between instances to connect related instances to facilitate representation learning (Nguyen et al., 2021a; Zhang and Ji, 2021). However, as the instance dependency graphs in previous work are only created manually using some heuristics, e.g., connecting instances that share an entity mention or event trigger (Nguyen et al., 2021a), they might be suboptimal for a given dataset and

[*] Work done at Raytheon BBN Technologies (prior to joining AWS AI).

Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, pages 9349–9360
December 7-11, 2022 ©2022 Association for Computational Linguistics

hinder further performance improvement for IE.

Consequently, to improve representation enrichment with information from related instances for JointIE, our work proposes to automatically learn cross-instance dependency graphs for IE tasks from data. To enable maximal flexibility, we explore a fully connected graph between all task instances in a sentence where a dependency weight is assigned to each edge to quantify the relatedness between two instances. In our method, we argue that dependency weights between task instances should be computed over multiple sources of information to produce optimal and comprehensive dependency graphs. To this end, motivated by the encoding of different linguistic structures (e.g., semantics, syntax) in the layers of pre-trained language models (PLMs), e.g., BERT (Devlin et al., 2019; Jawahar et al., 2019), we propose to leverage the representations of instances at different layers of PLMs to compute dependency weights for the instances. In particular, given two instances for JointIE, their representation vectors at each layer of a PLM are consumed to produce a layer-specific dependency weight, which will be combined across layers to obtain an overall weight for our dependency graph. Graph Convolutional Networks (GCNs) (Kipf and Welling, 2017; Nguyen and Grishman, 2018) will then be used to induce enriched representations for the instances based on the computed cross-instance dependency graph.

In addition, cross-type dependencies/patterns in JointIE systems characterize co-occurrences/co-relations of information types of different IE tasks (e.g., entity/event types and argument roles) in a single input sentence. For instance, in the ACE 2005 dataset (Walker et al., 2006), a *"Victim"* argument for an *"Attack"* event is likely to be the *"Victim"* argument for a *"Die"* event in the same sentence. Accordingly, previous JointIE models have leveraged cross-type dependencies either in the decoding phase, i.e., to form global type patterns/graphs to constrain the type prediction (Lin et al., 2020), or in the training phase, i.e., to form type dependency graphs to aid consistency regularization of golden and predicted types (Nguyen et al., 2021a). However, as in cross-instance dependencies, the dependency graphs between information types in IE in previous work are also designed manually, e.g., by linking types that are involved in the same instance for some IE task (Nguyen et al., 2021a). This is not desirable as manual designs might miss important cross-type patterns that cannot guarantee optimal performance for JointIE.

To this end, we propose to further learn cross-type dependencies/patterns from data to better support type predictions of JointIE instances. As such, we view each information type in our IE tasks as a binary random variable, which is 1 if the type appears in the sentence, and 0 otherwise. This formulation enables us to employ Bayesian structure learning algorithms to infer dependency structures from data. In particular, we propose to leverage the Chow-Liu algorithm (Chow and Liu, 1968) that measures mutual information between any two types (variables) in training data to learn a first-order dependency tree, aiming to approximate the underlying joint distribution of the information types (types) for JointIE. Afterward, the resulting Chow-Liu tree containing induced dependencies between information types will be used to generate global cross-type patterns for JointIE.

To incorporate the learned cross-type dependencies into the JointIE model, our goal is to leverage such global patterns to obtain additional features to further enrich the GCN-induced representations for type prediction. Our intuition is to treat the induced cross-type patterns as anchor knowledge to which the information types, representations, and dependencies of IE instances in a sentence should adhere to exploit consistency and improve predictions for JointIE in the data. To this end, for each learned cross-type pattern, we seek to compute a similarity score between the computed cross-instance dependency graph for an input sentence and the cross-type pattern that can be included into the representations for the instances to predict types. Accordingly, we propose to leverage random walk graph kernels (Gärtner et al., 2003; Feng et al., 2022) that facilitate similarity computation between two graphs (i.e., the cross-instance dependency graph and cross-type pattern) via counting common random walks on the graphs to enrich representations for JointIE. Finally, we evaluate the proposed model with induced cross-task and cross-type dependencies for JointIE in both monolingual and cross-lingual learning settings. Experimental results show that our model consistently outperforms strong baselines in all the settings across four different datasets and languages.

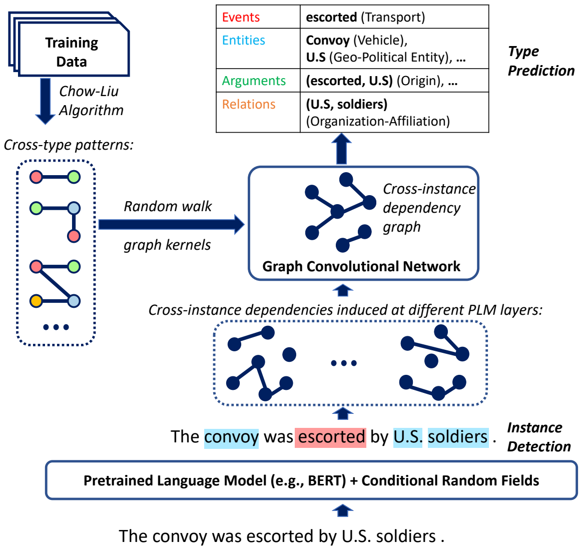

Training Data
Chow-Liu Algorithm
Cross-type patterns:
Random walk
graph kernels
Events | escorted (Transport)
Entities | Convoy (Vehicle), U.S (Geo-Political Entity), ...
Arguments | (escorted, U.S) (Origin), ...
Relations | (U.S, soldiers) (Organization-Affiliation)
Type Prediction
Cross-instance dependency graph
Graph Convolutional Network
Cross-instance dependencies induced at different PLM layers:
The convoy was escorted by U.S. soldiers .
Instance Detection
Pretrained Language Model (e.g., BERT) + Conditional Random Fields
The convoy was escorted by U.S. soldiers .

Figure 1: Overview of our JointIE model.

2 Model

There are four tasks in our IE pipeline, i.e., entity mention recognition (EMR), event detection (ED), event argument extraction (EAE), and relation extraction (RE). EMR and ED seek to identify word spans and types for entities (e.g., a *"Person"*) and events (e.g., an *"Attack"*) in text, respectively. On the other hand, EAE aims to identify whether each entity mention plays an argument role (e.g., an *"Attacker"*) in a given event mention. A special type *"Other-role"* is used to indicate that an entity does not play any role in a given event. For RE, the task is to determine if a relation (e.g., an *"Affiliation"* relation) exists between two given entity mentions. Similar to EAE, an special type *"Other-relation"* is used in RE to indicate no relation between two given entities. Joint information extraction (JointIE) is the joint task of EMR, ED, EAE, and RE (Lin et al., 2020; Nguyen et al., 2021a; Zhang and Ji, 2021), which aims to simultaneously predict entity mentions, event triggers, event arguments and relations for an input text in an end-to-end fashion.

Our proposed model (called *"DepIE"*) for JointIE consists of three main components: (i) Instance Detection, (ii) Cross-Instance Dependencies, and (iii) Cross-type Dependencies. Figure 1 presents an overview for our model.

2.1 Instance Detection

The first step in our model is to identify candidate instances for all the four IE tasks. In particular, candidate instances for EMR and ED involve spans of words for entity mentions and event triggers in text. For EAE, a candidate instance is formed by a pair of an event trigger span and an entity mention span. Similarly, we can obtain candidate instances for RE by pairing entity mention spans. Note that this step only performs candidate instance identification. Information types for the instances will be predicted in the next steps.

Event Triggers and Entity Mentions: Given an input sentence $\mathbf{w} = [w_1, \ldots, w_N]$ with N words, we employ a pretrained language model (PLM), e.g., RoBERTa (Liu et al., 2019), to produce a sequence of contextualized embeddings $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_N]$ for the words (using average of hidden vectors for word-pieces in the last layer of the PLM). The vector sequence $\mathbf{X}$ is then consumed by two different conditional random fields (CRFs) layers to predict two BIO tag sequences; each sequence aims to captures spans of event triggers (or entity mentions) for ED (or EMR). The negative log-likelihoods L_t and L_e returned by the CRFs for the ground-truth tag sequences of the spans for EMR and ED will then be included into the overall loss function. At test time, Viterbi algorithm is used to search for most probable tag sequences to find spans for event triggers $V_t = \{v_t\}$ and entity mentions $V_e = \{v_e\}$ (i.e., candidate instances) in the sentence. Each event trigger/entity mention is represented by a vector $\mathbf{v}_*$ ($* \in \{t, e\}$), computed via the average of contextualized embeddings for the words inside its corresponding spans v_*.

Event Arguments and Relations: While it is possible to use all pairs of entity mention and event trigger spans for the candidate instances of EAE and RE for type prediction, the large number of possible pairs will increase necessary computational resources. To this end, we first send the pairs into binary classifiers to determine if they are positive examples (i.e., corresponding to some actual types of interest for EAE and RE). In particular, to decide if an entity mention $v_e \in V_e$ plays any role with an event trigger $v_t \in V_t$, we concatenate their span vectors (i.e., $\mathbf{v}_e$ and $\mathbf{v}_t$) and feed the concatenation into a feed-forward network (FFN) with a sigmoid function in the end: $p_a = \sigma(FFN_a([\mathbf{v}_e; \mathbf{v}_t]))$. Here, the score $p_a \in (0, 1)$ represents the likelihood for v_e to be an argument of some role for v_t. Similarly, we can compute a score $p_r \in (0, 1)$ for all pairs of entity mentions $v_{e_1}, v_{e_2} \in V_e$ to estimate the likelihood that there exists a relation between the entity mentions. In the training process, we obtain the binary cross-entropy losses L_a and L_r computed with the probability scores p_a, p_r to include in the overall loss function. In test time, we employ a threshold of 0.5 for the scores p_a, p_r to determine positive pairs $V_a = \{v_a = (v_t, v_e)\}$ for event arguments and $V_r = \{v_r = (v_{e1}, v_{e2})\}$ for relations. Only positive pairs are retained for our next steps of type prediction. Finally, each positive event argument/relation is also represented by the average of representations of the involving event trigger and entity mention instances, called $\mathbf{v}_a$ and $\mathbf{v}_r$.

2.2 Cross-Instance Dependencies

Given the detected instances for the four IE tasks in $\mathbf{w}$, we aim to enrich the representation for each instance with information from other related instances to facilitate type prediction. As such, our model first learns a dependency graph $G^{inst} = (V, E)$ to capture the relatedness for the instances (called cross-instance dependency graph). In particular, the node set V of G^{inst} involves all the detected instances, i.e., $V = V_t \cup V_e \cup V_a \cup V_r$. To enable information flow across different instances, our edge set E will include an edge for each possible pair of instances in V; a weight α_{ij} will be assigned to each pair (v_i, v_j) to quantify the dependency between v_i and v_j in V.

To learn the dependency weights α_{ij}, our intuition is to exploit information from different sources (e.g., semantics, syntax) to ensure comprehensive coverage of relatedness aspects for JointIE. Motivated by different linguistic features encoded in different transformer layers of PLMs (Jawahar et al., 2019), we propose to treat each layer of BERT (with L layers) as a source of information. In particular, each word in the input sentence will be represented by L different embeddings returned by each layer of the PLM. In this way, for each node in V, we can obtain L different node representations computed at each layer of BERT (by averaging representations for word-pieces). Let $\mathbf{v}_i^l, \mathbf{v}_j^l$ be the representations for the nodes $v_i, v_j \in V$ at layer l of the PLM. The dependency weight $\alpha_{ij}^l \in (0, 1)$ between the instance nodes v_i, v_j at layer l of BERT is computed by: $\alpha_{ij}^l = FFN_\sigma^l([\mathbf{v}_i^l; \mathbf{v}_j^l])$, where FFN_σ^l is a feed-forward network with a sigmoid function in the end.

To this end, each instance $v_i \in V$ is associated with L sets of weights $\{\alpha_{ij}^l\}$ capturing its dependencies on the other instances according to L different sources of information from BERT. The importance of the l-th information source to representation learning of v_i is then measured by sending its l-th representation $\mathbf{v}_i^l$ to a feed-forward network $FFN_{src}(\mathbf{v}_i^l)$. Afterward, we normalize the layer-specific importance scores for v_i across layers with softmax, leading to $s_i^l = \text{softmax}_l(FFN_{src}(\mathbf{v}_i^{1:L}))$. The dependency weight between v_i and v_j in our cross-instance graph is then determined via: $\alpha_{ij} = \sum_l s_i^l \alpha_{ij}^l$.

Finally, the induced dependency graph with weights α_{ij} is used to enhance the representations for $v_i \in V$ via a Graph Convolutional Network (GCN) (Kipf and Welling, 2017; Nguyen and Grishman, 2018) with K layers:

$$\mathbf{h}_i^k = \text{ReLU}\left(\frac{\sum_{v_j \in V} \alpha_{ij} \mathbf{W}^k \mathbf{h}_j^{k-1} + \mathbf{b}^k}{\sum_{v_j \in V} \alpha_{ij}}\right), 1 \leq k \leq K$$

where $\mathbf{h}_i^k$ is the representation for v_i at the k-th layer of GCN ($\mathbf{h}_i^0 = \mathbf{v}_i$). For convenience, let $\mathbf{h}_i$ be the representation for the instance v_i at the final layer of the GCN, i.e., $\mathbf{h}_i = \mathbf{h}_i^K$.

2.3 Cross-Type Dependencies

As discussed in the introduction, to further improve the representations for the instances v_i for type prediction, our method proposes to induce global dependencies between information types for different IE tasks (called cross-type dependencies) from data and use them as knowledge to generate additional features for instance representations.

Cross-type Dependency Induction: For convenience, let T be the set of all information types for our four IE tasks, i.e., including entity types, event types, event argument roles, and relations. To infer dependencies/patterns between the types in T, our goal is to leverage their co-occurrences in the sentences of training data for the computation. As such, we consider the information types in T as random variables and leverage the well-known Chow-Liu algorithm (Chow and Liu, 1968) in Bayesian structure learning to find meaningful relationships/patterns among the types. The Chow-Liu algorithm approximates the underlying joint distribution of random variables by finding a first-order dependency tree among the variables (i.e., tree nodes correspond to the variables).

Let $X_i \in \{0, 1\}$ be the binary random variable for the information type $t_i \in T$ where $X_i = 1$ if there exists one instance with type t_i in the current sentence, and $X_i = 0$ otherwise. The algorithm then computes mutual information (MI) scores between any two random variables X_i, X_j via:

$$I(X_i, X_j) = \sum_{x_i, x_j \in \{0,1\}} \hat{P}(x_i, x_j) \log \frac{\hat{P}(x_i, x_j)}{\hat{P}(x_i)\hat{P}(x_j)}$$

where $\hat{P}(x_i, x_j) = \frac{count(X_i = x_i, X_j = x_j)}{M}$ is the empirical joint distribution between X_i and X_j computed by counting across training data (M is the total number of sentences in the training data). Similarly, we can compute the marginal distributions $\hat{P}(x_i)$ and $\hat{P}(x_j)$. Afterwards, we construct a cross-type dependency tree G^{ctp} for information types as the spanning tree over the random variables that achieves maximum sum of the MI scores. The maximum spanning tree can be solved via Kruskal (Kruskal, 1956) or Prim (Prim, 1957) algorithms.

To make it more manageable, we collect the set of connected sub-graphs (i.e., trees) U that have at least two nodes and less than n nodes in G^{ctp} ($2 \le n \le |T|$ is a hyper-parameter) to serve as the global cross-type patterns/dependencies induced by our method for JointIE.

Feature Generation with Graph Kernels: Using the induced cross-type patterns $G_d^{ctp} \in U$ from data as anchor knowledge, we expect the information types, instance representations, and instance dependencies in an input sentence **w** to follow the patterns to exploit consistency in the data. In particular, instance representations and dependencies in an input sentence will have higher quality for type prediction if they are more similar to the induced cross-type patterns from data. Accordingly, we propose to leverage similarity scores between the cross-instance dependency graph for **w** and the cross-type patterns in U as additional features to improve representations for JointIE. Here, we can employ the cross-instance dependency graph G^{inst} with dependency weights α_{ij} computed in the previous step for the feature computation.

As such, to compute the similarity between G^{inst} and G_d^{ctp}, we propose to employ random walk graph kernels (Gärtner et al., 2003) that can facilitate similarity measurement between two graphs with different number of nodes. In particular, the random walk kernel is computed by counting the number of common random walks on the two graphs, which has been shown to be equivalent to performing a random walk on the direct product of the graphs (Vishwanathan et al., 2006). This enables the p-step random walk kernel between two graphs G_1 and G_2 to be efficiently computed via: (Vishwanathan et al., 2006; Feng et al., 2022):

$$K_p(G_1, G_2) = \sum_{i,j} \left[(\mathbf{V}_1\mathbf{V}_2^T) \odot (\mathbf{A}_1^p \mathbf{V}_1 (\mathbf{A}_2^p \mathbf{V}_2)^T)\right]_{ij}$$

where $\mathbf{V}_1$ and $\mathbf{V}_2$ are the node embedding matrices for the node sets; $\mathbf{A}_1$ and $\mathbf{A}_2$ are adjacency matrices for the graphs G_1 and G_2 respectively; $\odot$ is the element-wise product, and $\mathbf{A}_*^p$ is the p-th power of the matrix $\mathbf{A}_*$ ($* \in \{1, 2\}$).

To adapt this random walk kernel for G^{inst} and G_d^{ctp}, we can obtain the adjacency matrix $\mathbf{A}^{inst}$ for G^{inst} from the dependency weights α_{ij}, i.e., $\mathbf{A}_{ij}^{inst} = \alpha_{ij}$. The node embedding matrix $\mathbf{V}^{inst}$ for G^{inst} can leverage the GCN-induced vectors by setting the i-th row of $\mathbf{V}^{inst}$ to $\mathbf{h}_i$ for instance $v_i \in V$. Also, for each induced cross-type pattern/tree $G_d^{ctp} \in U$, we can use its binary adjacency matrix $\mathbf{A}_d^{ctp}$ for the kernel computation. Its node embedding matrix $\mathbf{V}_d^{ctp}$ will be produced by looking up the corresponding types in a type embedding matrix $\mathbf{T}$ for all types in T. In our method, $\mathbf{T}$ is initialized randomly so its embedding dimension is equal to those for the instance representation $\mathbf{h}_i$. In this way, we can compute a kernel-based similarity score $ks_d = K_p(G^{inst}, G_d^{ctp})$ between the cross-instance dependency graph G^{inst} and each cross-type pattern in U. Finally, the concatenation of such similarity scores, i.e., $\mathbf{m}^{ctp} = [ks_1, ks_2, \ldots, ks_{|U|}]$, can be used to provide additional global features for the instance representations for type predictions. Note that in this way, our cross-type patterns can support both training and test phases for JointIE models. This is in contrast to previous methods that can only utilize manually designed patterns in either training (e.g., FourIE) or decoding (e.g., OneIE) phase.

Training: To predict type for each instance $v_i \in V$, we compute an overall representation vector $\mathbf{r}_i$ for v_i by concatenating its GCN-induced representation $\mathbf{h}_i$ and the global features $\mathbf{m}^{ptn}$: $\mathbf{r}_i = FFN_{pred}(\text{concat}(\mathbf{h}_i, \mathbf{m}^{ptn}))$. Here, FFN_{pred} is a feed-forward network to ensure that $\mathbf{r}_i$ has the same dimension as the type embeddings $\mathbf{T}$. The type distribution v_i is then estimated by normalizing the similarity of $\mathbf{r}_i$ and the type embeddings: $\hat{\mathbf{y}}_i = \text{softmax}(\mathbf{r}_i\mathbf{t}^T | \mathbf{t} \in \mathbf{T}_i)$ where $\mathbf{T}_i$ is the set of embeddings for all possible types T_i for v_i in T. The negative log-likelihood of the ground-truth types t_i is then used to train our model: $L_{cls} = -\sum_{v_i \in V} \log(\hat{\mathbf{y}}_i[t_i])$. In summary, the overall training loss for our model is: $L = L_t + L_e + L_a + L_r + L_{cls}$.

3 Experiments

Datasets: Following previous work (Lin et al., 2020; Nguyen et al., 2021a), we conduct experiments on four datasets with different languages, i.e., ACE05-E+ (English), ACE05-CN (Chinese), ACE05-AR (Arabic), and ERE-ES (Spanish). The three ACE05 datasets are created by the Automatic Content Extraction program (Walker et al., 2006) with 33 event types, 7 entity types, 6 relation types, and 22 argument roles; and the ERE-ES dataset is from the Deep Exploration and Filtering of Text program (DEFT) (Song et al., 2015) with a similar schema to ACE05 datasets. For a fair comparison, we use the same preprocessing and train/dev/test splits for ACE05-E+, ACE05-CN, and ERE-ES as provided by prior work (Lin et al., 2020; Nguyen et al., 2021a). The ACE05-AR dataset does not have a standard split for JointIE so we follow the data split by (M'hamdi et al., 2019) for ETD in Arabic and apply the same preprocessing code from previous work (Lin et al., 2020) to produce the train/dev/test sets for ACE05-AR. Additionally, we perform experiments on the IARPA BETTER program[1]'s Basic Event Extraction datasets, which feature 118 event types, 3 mention types, and 3 argument roles. The BETTER-EN dataset is obtained by respectively combining the official training, development, and test parts of Phase 1, 2, and 3 English data. For the BETTER-FA dataset, we randomly split the Phase 2 Farsi evaluation data into training, development, and test portions with a ratio of 70/15/15 as no standard split is provided. Statistics for all the datasets are shown in Table 2.

Hyper-Parameters: For the PLMs, we use RoBERTa large (Liu et al., 2019) and its multilingual version XLM-RoBERTa large (Conneau et al., 2020) for English and non-English datasets respectively. We tune hyper-parameters for our model on ACE05-E+ development data and apply the best hyper-parameters to the other datasets for consistency. In particular, we select: 5e-6 for learning rate with Adam optimizer; 10 for batch size; 300 for the hidden vector sizes for all the feed-forward networks and the GCN model; 2 for the number of layers for the feed-forward and GCN networks; $n = 4$ for the sizes of cross-type patterns in U; and $p = 2$ for the kernel computation. The model performance is obtained by averaging over three runs with different random seeds.

Baselines: We compare our method (i.e., **DepIE**) with recent models that jointly perform our four IE tasks, including **OneIE** (Lin et al., 2020), **AMRIE** (Zhang and Ji, 2021), and **FourIE** (Nguyen et al., 2021a). FourIE is the current state-of-the-art method for JointIE. Among models, OneIE, FourIE, and our model DepIE are language-agnostic so they can be directly applied to non-English datasets. In contrast, AMRIE is only designed for English as it requires an English AMR parser. To be comprehensive, we also consider recent event extraction methods, i.e., **Text2event** (Lu et al., 2021), **DEGREE-E2E** (Hsu et al., 2021), **Query&Extract** (Wang et al., 2022), **GTEE-DYNPREF** (Liu et al., 2022), which perform only ETD and EAE.

Monolingual Performance: We first compare the models in monolingual settings across the four datasets in Tables 1 and 3 where models are trained and tested on data of the same language. As can be seen, our model performs significantly better than the baselines across the datasets. Among the four IE tasks, the EAE and RE tasks appear to gain largest performance improvements. Further, as the improvements are consistent across languages, it highlights the portability to different languages of the induced cross-instance and cross-task dependencies in our proposed model for JointIE.

Crosslingual Performance: To further investigate the cross-lingual generalization of the JointIE models, we compare OneIE, FourIE, and DepIE in the cross-lingual transfer learning settings where the models are trained on training data of English datasets and evaluated on the test data of the other languages. As shown in Table 4, our model DepIE is still the best performer in the crosslingual settings over different tasks and test languages. The performance improvement is significant on almost all tasks ($p < 0.01$), thus demonstrating language-invariant advantages of our designed cross-task dependencies for JointIE. In addition, we note that this is the first comprehensive evaluation of JointIE models in cross-lingual transfer learning. As the performance of the current models is still not satisfactory, it emphasizes the challenges of JointIE with cross-lingual transfer learning and call for future research efforts in this important direction.

Ablation Study: To study the impact of each pro-

[1] https://www.iarpa.gov/index.php/research-programs/better

Model	ACE05-E+ (English)				ACE05-CN (Chinese)				ACE05-AR (Arabic)				ERE-ES (Spanish)			
	Ent	Rel	Trg	Arg	Ent	Rel	Trg	Arg	Ent	Rel	Trg	Arg	Ent	Rel	Trg	Arg
Text2event	-	-	71.8	54.4	-	-	-	-	-	-	-	-	-	-	-	-
DEGREE-E2E	-	-	71.7	56.8	-	-	-	-	-	-	-	-	-	-	-	-
Query&Extract	-	-	73.6	55.1	-	-	-	-	-	-	-	-	-	-	-	-
GTEE-DYNPREF	-	-	74.3	54.7	-	-	-	-	-	-	-	-	-	-	-	-
OneIE	90.8	60.4	72.5	56.3	88.5	64.9	67.3	54.8	81.2	59.0	56.6	37.2	83.7	57.5	58.3	42.5
AMRIE	91.0	62.8	72.7	57.7	-	-	-	-	-	-	-	-	-	-	-	-
FourIE	91.1	63.1	72.8	58.3	88.8	66.0	69.1	57.5	81.7	61.4	57.9	42.1	83.8	59.0	63.4	45.1
DepIE (Ours)	**91.7**	**64.9**	**74.6**	**61.2**	**89.2**	**68.3**	**74.3**	**60.0**	**82.7**	**63.5**	**63.1**	**46.4**	**86.5**	**61.2**	**65.9**	**51.9**

Table 1: Monolingual performance on test data of the datasets. *"Ent"*, *"Rel"*, *"Trg"*, and *"Arg"* indicate F1 scores for identification and classification of entity mentions, relations, event triggers, and arguments respectively. All results are reported by the original papers or produced by running the official code. All JointIE models use large RoBERTa. Underlined numbers indicate that *DepIE* is significantly better than the baselines ($p < 0.01$).

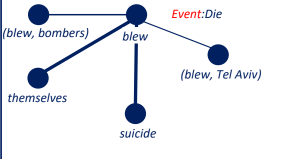
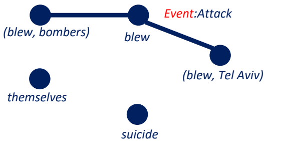
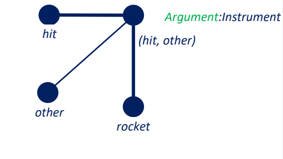
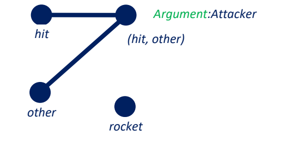
Example	DepIE	FourIE
In the January attack, two Palestinian suicide bombers blew themselves up in central Tel Aviv, killing 23 other people. ***Analysis***: *DepIE can successfully predict "blew" as a "Die" event trigger due to the recognized connections with "suicide" and "themselves" while FourIE fails to do so.*	 Event:Die (blew, bombers) blew (blew, Tel Aviv) themselves suicide	 Event:Attack (blew, bombers) blew (blew, Tel Aviv) themselves suicide
A second rocket landed in farmlands and the other hit a house inside the refugee camp, ... ***Analysis***: *DepIE can successfully predict "other" as an "Instrument" for the event trigger "hit" due to its ability to connect to the important related instance "rocket" while FourIE fails to do so.*	 Argument:Instrument hit (hit, other) other rocket	

Figure 2: Some task instances along with their dependency connections produced by DepIE and FourIE.

Datasets	Split	#sents	#ents	#rels	#events
ACE05-E+	Train	19,240	47,525	7,152	4,419
	Dev	902	3,422	728	468
	Test	676	3,673	802	424
ACE05-CN	Train	6,841	29,657	7,934	2,926
	Dev	526	2,250	596	217
	Test	547	2,388	672	190
ACE05-AR	Train	1,915	28,113	4,063	1,198
	Dev	108	1,892	275	112
	Test	152	2,495	374	169
ERE-ES	Train	7,067	11,839	1,698	3,272
	Dev	556	886	120	210
	Test	546	811	108	269
BETTER-EN	Train	5,617	18,815	-	16,594
	Dev	1,163	3,958	-	3,177
	Test	1,173	3,707	-	3,311
BETTER-FA	Train	2,932	11,612	-	10,100
	Dev	592	2,377	-	2,061
	Test	658	2,468	-	2,054

Table 2: Dataset statistics. **#sents**, **#ent**, **#rels**, and **#events** represent the numbers of sentences, entity mentions, relations, and events respectively.

Datasets	Task	OneIE	FourIE	DepIE (Ours)
BETTER-EN (English)	Ent	75.1	75.3	**76.5**
	Trg	63.6	63.9	**65.6**
	Arg	62.4	64.5	**65.6**
BETTER-FA (Farsi)	Ent	65.1	65.7	**66.5**
	Trg	57.0	57.6	**59.1**
	Arg	55.2	56.3	**58.1**

Table 3: Monolingual performance (F1 scores) on test data of BETTER datasets.

posed component for DepIE, Table 5 evaluates the ablated models over ACE05-E+ development data.

In particular, for cross-instance dependencies, we first remove the cross-instance dependency graph from DepIE. The ablated model *"- cross-instance"* shows significant performance drops across all the four IE tasks, demonstrating the importance of the cross-instance dependency component to our model. In addition, we evaluate a simplified version of this component where a single source of information is used to induce dependencies between instances. Particularly, the cross-instance dependency weights α_{ij} in this case are computed with only the last layer of the PLM instead of all the layers. As the performance of the ablated model *"+single-source graph"* is substantially worse than the full model, it confirms the benefits of using multiple information sources from PLM to compute cross-instance dependencies for FourIE. Moreover, we replace our induced dependency weights for instances with the heuristic-based dependency weights produced by the best baseline model FourIE (i.e., $\alpha_{ij} = 1$ if instances v_i and v_j share an event trigger or entity mention). The inferior performance of the resulting model

Test Data	Task	OneIE	FourIE	DepIE (Ours)
ACE05-CN	Ent	70.2	70.8	**71.8**
	Rel	31.1	32.6	**35.7**
	Trg	58.4	60.5	**62.1**
	Arg	37.9	39.2	**41.5**
ACE05-AR	Ent	64.2	65.4	**66.5**
	Rel	27.1	30.6	**31.7**
	Trg	35.4	36.9	**40.6**
	Arg	25.0	26.5	**28.0**
ERE-ES	Ent	75.5	76.5	**76.6**
	Rel	27.7	28.6	**33.0**
	Trg	45.3	47.0	**49.9**
	Arg	34.2	35.4	**37.4**
BETTER-FA	Ent	74.1	74.2	**74.8**
	Trg	56.5	57.3	**58.7**
	Arg	59.8	61.7	**63.0**

Table 4: Cross-lingual performance (F1 scores) on test data of non-English datasets. For the BETTER-FA setting, the models are trained on training data of BETTER-EN only. For the other settings, only training data of ACE05-E+ is used for training.

Models	**ACE05-E+**			
	Ent	Rel	Trg	Arg
DepIE	**89.1**	**65.6**	**73.3**	**65.3**
- cross-instance	87.4	62.7	71.7	62.0
+ single-source graph	88.6	64.3	72.7	63.7
+ heuristic graph	88.1	63.1	72.2	62.9
- GCN	88.3	63.8	72.4	63.1
- cross-type	88.2	64.1	72.0	64.1
+ naive cross-type	87.8	63.5	71.6	63.7
+ cosine similarity	88.4	64.5	72.8	64.3
+ type regularization	88.2	64.6	72.4	64.5
+ global features	87.7	63.1	72.0	64.0

Table 5: Model performance (F1) of ablated models.

“+heuristic graph” compared to *“+single-source graph”* and DepIE strongly indicates the strength of automatically learned dependency graphs for JointIE. Finally, we report the performance of DepIE where the GCN model is removed while still preserving the cross-instance and cross-type dependencies (i.e., *“- GCN”*). As such, the contextualized embeddings $\mathbf{x}_i$ will replace the GCN-induced vectors $\mathbf{h}_i$ in the computation. It is clear from the table that the GCN model is necessary for DepIE as *“- GCN”* has significantly worse performance.

Next, we study the effect of the cross-type dependency component for DepIE. As shown in the table, removing cross-type dependencies from DepIE (i.e., *“- cross-type”*) significantly hurts model performance. To understand the benefit of the Chow-Liu algorithm, we examine a simpler method to produce the cross-type dependency graph G^{ctp} where two information types in T are connected if they are both expressed in a sentence in training data. The resulting model (i.e., *“+ naive cross-type”*) performs much poorer than our full model with

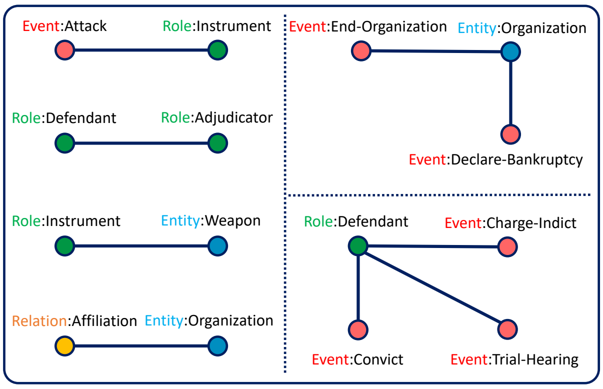


Figure 3: Cross-type patterns learned DepIE on ACE05-E+. Blue, red, green, and orange circles represent entity, event, argument role, and relation types respectively.

the Chow-Liw tree. To investigate the effectiveness of the random walk kernels, we examine a similar method to the type dependency regularization in FourIE to compute the similarity between the cross-instance graph G^{inst} and the cross-type patterns G_d^{cpt} for the global features $\mathbf{m}^{cpt}$. In particular, we use a GCN model to consume the graphs G^{inst} and G_d^{cpt} along with their node embeddings; the resulting vectors for each graph are then max-pooled to obtain a representation vector for the graph. The similarity between the two graphs is then computed via the cosine similarity between their representations. As the corresponding model *“+ cosine similarity”* is worse than the full model over different tasks, it demonstrates the necessity of the random walk kernels for DepIE.

Finally, we remove the cross-type dependency component (i.e., with Chow-Liu and graph kernels) and integrate alternative methods to generate and apply cross-type dependencies from previous JointIE methods into DepIE, i.e., the type regularization in FourIE for training or the global type features for decoding in OneIE. Both the models *“+type regularization”* and *“+global features”* in Table 5 observe large decreased performance, further confirming the benefit of the cross-type dependency components for JointIE in DepIE.

Analysis: To understand the effect of the cross-instance dependency graph learned by DepIE compared to the heuristic-based dependency graph produced by FourIE, we examine examples on the ACE05-E+ development data for which DepIE can have correct predictions while FourIE fails to do so. Figure 2 presents some examples of this type. As can be seen, by computing dependency weights for all possible pairs of instances, DepIE can discover important related instances that do not share any entity mentions/event triggers with the instance of interest (e.g., the related instance *"suicide"* for *"blew"*), thus allow DepIE to correct the wrong predictions in FourIE to improve the performance.

Finally, Figure 3 presents some cross-type patterns learned DepIE. We observe that 3-node and 4-node patterns can capture subtle structures between information types for JointIE (e.g., the *"Charge-Indict"*, *"Convict"*, and *"Trial-Hearring"* event types and the *"Defendant"* argument role).

4 Related Work

IE tasks have been performed jointly to capture dependency between the tasks via feature engineering (Roth and Yih, 2004; Yu and Lam, 2010; Li et al., 2013; Yang and Mitchell, 2016) or deep learning (Nguyen et al., 2016; Zheng et al., 2017; Bekoulis et al., 2018; Luan et al., 2019) methods. However, most previous work only jointly solves two or three IE tasks (Nguyen and Nguyen, 2019; Lu et al., 2021). Recently, there have been growing interest in performing all the four IE tasks jointly (i.e., JointIE) (Wadden et al., 2019; Zhang and Ji, 2021; Nguyen et al., 2022) to exploit manually designed dependency graphs for IE instances (Nguyen et al., 2021a) or handcrafted global features for information types (Lin et al., 2020). Our work is different from previous JointIE models as we learn cross-instance and cross-type dependencies from data to provide better structures for representation learning. Finally, we note that our cross-type dependency component is related to structure learning methods for Bayesian networks (Eaton and Murphy, 2012; Banerjee and Ghosal, 2015; Scutari et al., 2019) and graph kernels to compute graph similarity (Gärtner et al., 2003; Vishwanathan et al., 2006; Shervashidze et al., 2009; Kondor and Pan, 2016a; Feng et al., 2022). However, these approaches have not been explored for JointIE.

5 Conclusion

We present a novel model to jointly solve four IE tasks (EMR, ETD, EAE, and RE). Our model learns cross-instance dependencies through different layers of a PLM and cross-type dependencies via the Chow-Liu algorithm. The cross-task dependencies are exploited via GCNs and random walk kernels to improve representation learning. Extensive experiments demonstrate the state-of-the-art performance of our model across four datasets with different languages and settings. In the future, we plan to extend our model to include more IE tasks.

Limitations

In this work we propose a novel model to jointly solve four tasks in information extraction, i.e., entity mention recognition, event trigger detection, event argument extraction, and relation extraction (JointIE). Although our experiments demonstrate the effectiveness of the proposed method, there are still some limitations that can be improved in future work. First, other graph kernels to compute graph similarity such as subgraph matching kernels (Kriege and Mutzel, 2012) or multiscale Laplacian graph kernels (Kondor and Pan, 2016b) are not yet explored in the current work. Future work can explore such alternatives for graph kernels to improve the effectiveness of graph comparison for representation learning. Second, the Chow-Liu algorithm employed by our model is a popular method in Bayesian structure learning; however, recent structure learning methods such as ordering-based search (Teyssier and Koller, 2005) and integer linear programming (Cussens et al., 2017) are not evaluated in our work. These methods can be considered to improve the learning of cross-type dependencies in our work.

Acknowledgement

This research has been supported by the Army Research Office (ARO) grant W911NF-21-1-0112 and the NSF grant CNS-1747798 to the IUCRC Center for Big Learning. 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

Sayantan Banerjee and Subhashis Ghosal. 2015. Bayesian structure learning in graphical models. *Journal of Multivariate Analysis*, 136:147–162.

Giannis Bekoulis, Johannes Deleu, Thomas Demeester, and Chris Develder. 2018. Adversarial training for multi-context joint entity and relation extraction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2830–2836, Brussels, Belgium. Association for Computational Linguistics.

Yubo Chen, Liheng Xu, Kang Liu, Daojian Zeng, and Jun Zhao. 2015. Event extraction via dynamic multi-pooling convolutional neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 167–176, Beijing, China. Association for Computational Linguistics.

CKCN Chow and Cong Liu. 1968. Approximating discrete probability distributions with dependence trees. *IEEE transactions on Information Theory*, 14(3):462–467.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

James Cussens, Matti Järvisalo, Janne H. Korhonen, and Mark Bartlett. 2017. Bayesian network structure learning with integer programming: Polytopes, facets and complexity (extended abstract). In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pages 4990–4994.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Xinya Du and Claire Cardie. 2020. Event extraction by answering (almost) natural questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 671–683, Online. Association for Computational Linguistics.

Daniel Eaton and Kevin Murphy. 2012. Bayesian structure learning using dynamic programming and mcmc. *arXiv preprint arXiv:1206.5247*.

Aosong Feng, Chenyu You, Shiqiang Wang, and Leandros Tassiulas. 2022. Kergnns: Interpretable graph neural networks with graph kernels. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Thomas Gärtner, Peter Flach, and Stefan Wrobel. 2003. On graph kernels: Hardness results and efficient alternatives. In *Learning theory and kernel machines*, pages 129–143. Springer.

I Hsu, Kuan-Hao Huang, Elizabeth Boschee, Scott Miller, Prem Natarajan, Kai-Wei Chang, Nanyun Peng, et al. 2021. Degree: A data-efficient generative event extraction model. *arXiv preprint arXiv:2108.12724*.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does BERT learn about the structure of language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657, Florence, Italy. Association for Computational Linguistics.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *Proceedings of the 5th International Conference on Learning Representations*.

Risi Kondor and Horace Pan. 2016a. The multiscale laplacian graph kernel. *Advances in neural information processing systems*, 29.

Risi Kondor and Horace Pan. 2016b. The multiscale laplacian graph kernel. In *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc.

Nils M. Kriege and Petra Mutzel. 2012. Subgraph matching kernels for attributed graphs. In *ICML*.

Joseph B Kruskal. 1956. On the shortest spanning subtree of a graph and the traveling salesman problem. *Proceedings of the American Mathematical society*, 7(1):48–50.

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 *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Fayuan Li, Weihua Peng, Yuguang Chen, Quan Wang, Lu Pan, Yajuan Lyu, and Yong Zhu. 2020. Event extraction as multi-turn question answering. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 829–838, Online. Association for Computational Linguistics.

Qi Li, Heng Ji, and Liang Huang. 2013. Joint event extraction via structured prediction with global features. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 73–82, Sofia, Bulgaria. Association for Computational Linguistics.

Ying Lin, Heng Ji, Fei Huang, and Lingfei Wu. 2020. A joint neural model for information extraction with global features. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7999–8009, Online. Association for Computational Linguistics.

Xiao Liu, Heyan Huang, Ge Shi, and Bo Wang. 2022. Dynamic prefix-tuning for generative template-based event extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5216–5228, Dublin, Ireland. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Yaojie Lu, Hongyu Lin, Jin Xu, Xianpei Han, Jialong Tang, Annan Li, Le Sun, Meng Liao, and Shaoyi Chen. 2021. Text2Event: Controllable sequence-to-structure generation for end-to-end event extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2795–2806, Online. Association for Computational Linguistics.

Yi Luan, Dave Wadden, Luheng He, Amy Shah, Mari Ostendorf, and Hannaneh Hajishirzi. 2019. A general framework for information extraction using dynamic span graphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3036–3046, Minneapolis, Minnesota. Association for Computational Linguistics.

Meryem M'hamdi, Marjorie Freedman, and Jonathan May. 2019. Contextualized cross-lingual event trigger extraction with minimal resources. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 656–665, Hong Kong, China. Association for Computational Linguistics.

Minh Van Nguyen, Viet Lai, and Thien Huu Nguyen. 2021a. Cross-task instance representation interactions and label dependencies for joint information extraction with graph convolutional networks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 27–38, Online. Association for Computational Linguistics.

Minh Van Nguyen, Bonan Min, Franck Dernoncourt, and Thien Nguyen. 2022. Joint extraction of entities, relations, and events via modeling inter-instance and inter-label dependencies. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4363–4374, Seattle, United States. Association for Computational Linguistics.

Minh Van Nguyen, Tuan Ngo Nguyen, Bonan Min, and Thien Huu Nguyen. 2021b. Crosslingual transfer learning for relation and event extraction via word category and class alignments. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5414–5426, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Thien Huu Nguyen, Kyunghyun Cho, and Ralph Grishman. 2016. Joint event extraction via recurrent neural networks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 300–309, San Diego, California. Association for Computational Linguistics.

Thien Huu Nguyen and Ralph Grishman. 2015. Event detection and domain adaptation with convolutional neural networks. In *The 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*.

Thien Huu Nguyen and Ralph Grishman. 2018. Graph convolutional networks with argument-aware pooling for event detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Trung Minh Nguyen and Thien Huu Nguyen. 2019. One for all: Neural joint modeling of entities and events. In *Proceedings of the Association for the Advancement of Artificial Intelligence (AAAI)*.

Amir Pouran Ben Veyseh, Viet Lai, Franck Dernoncourt, and Thien Huu Nguyen. 2021. Unleash GPT-2 power for event detection. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6271–6282, Online. Association for Computational Linguistics.

Robert Clay Prim. 1957. Shortest connection networks and some generalizations. *The Bell System Technical Journal*, 36(6):1389–1401.

Dan Roth and Wen-tau Yih. 2004. A linear programming formulation for global inference in natural language tasks. In *Proceedings of the Eighth Conference on Computational Natural Language Learning (CoNLL-2004) at HLT-NAACL 2004*, pages 1–8, Boston, Massachusetts, USA. Association for Computational Linguistics.

Marco Scutari, Catharina Elisabeth Graafland, and José Manuel Gutiérrez. 2019. Who learns better bayesian network structures: Accuracy and speed of structure learning algorithms. *International Journal of Approximate Reasoning*, 115:235–253.

Nino Shervashidze, SVN Vishwanathan, Tobias Petri, Kurt Mehlhorn, and Karsten Borgwardt. 2009. Efficient graphlet kernels for large graph comparison. In *Artificial intelligence and statistics*, pages 488–495. PMLR.

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

Marc Teyssier and Daphne Koller. 2005. Ordering-based search: A simple and effective algorithm for learning bayesian networks. In *Proceedings of the Twenty-First Conference on Uncertainty in Artificial Intelligence*, UAI'05, page 584–590, Arlington, Virginia, USA. AUAI Press.

Amir Pouran Ben Veyseh, Tuan Ngo Nguyen, and Thien Huu Nguyen. 2020. Graph transformer networks with syntactic and semantic structures for event argument extraction. In *Proceedings of the Findings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP Findings)*.

SVN Vishwanathan, Karsten M Borgwardt, Nicol N Schraudolph, et al. 2006. Fast computation of graph kernels. In *NIPS*, volume 19, pages 131–138. Citeseer.

David Wadden, Ulme Wennberg, Yi Luan, and Hannaneh Hajishirzi. 2019. Entity, relation, and event extraction with contextualized span representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5784–5789, Hong Kong, China. Association for Computational Linguistics.

Christopher Walker, Stephanie Strassel, Julie Medero, and Kazuaki Maeda. 2006. Ace 2005 multilingual training corpus. In *Technical report, Linguistic Data Consortium*.

Sijia Wang, Mo Yu, Shiyu Chang, Lichao Sun, and Lifu Huang. 2022. Query and extract: Refining event extraction as type-oriented binary decoding. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 169–182, Dublin, Ireland. Association for Computational Linguistics.

Bishan Yang and Tom M. Mitchell. 2016. Joint extraction of events and entities within a document context. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 289–299, San Diego, California. Association for Computational Linguistics.

Xiaofeng Yu and Wai Lam. 2010. Jointly identifying entities and extracting relations in encyclopedia text via a graphical model approach. In *Coling 2010: Posters*, pages 1399–1407, Beijing, China. Coling 2010 Organizing Committee.

Zixuan Zhang and Heng Ji. 2021. Abstract Meaning Representation guided graph encoding and decoding for joint information extraction. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 39–49, Online. Association for Computational Linguistics.

Suncong Zheng, Feng Wang, Hongyun Bao, Yuexing Hao, Peng Zhou, and Bo Xu. 2017. Joint extraction of entities and relations based on a novel tagging scheme. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1227–1236, Vancouver, Canada. Association for Computational Linguistics.