Modeling Instance Interactions for Joint Information Extraction with Neural High-Order Conditional Random Field

Zixia Jia[1,2]*, **Zhaohui Yan**[2], **Wenjuan Han**[3], **Zilong Zheng**[1]†, **Kewei Tu**[2]†
[1] Beijing Institute for General Artificial Intelligence (BIGAI), Beijing, China
[2] ShanghaiTech University, Shanghai, China
[3] Beijing Jiaotong University, Beijing, China
{jiazixia,zlzheng}@bigai.ai
{yanzhh,tukw}@shanghaitech.edu.cn, wjhan@bjtu.edu.cn

Abstract

Prior works on joint Information Extraction (IE) typically model instance (*e.g.*, event triggers, entities, roles, relations) interactions by representation enhancement, type dependencies scoring, or global decoding. We find that the previous models generally consider binary type dependency scoring of a pair of instances, and leverage local search such as beam search to approximate global solutions. To better integrate cross-instance interactions, in this work, we introduce a joint IE framework (CRFIE) that formulates joint IE as a high-order Conditional Random Field. Specifically, we design binary factors and ternary factors to directly model interactions between not only a pair of instances but also triplets. Then, these factors are utilized to jointly predict labels of all instances. To address the intractability problem of exact high-order inference, we incorporate a high-order neural decoder that is unfolded from a mean-field variational inference method, which achieves consistent learning and inference. The experimental results show that our approach achieves consistent improvements on three IE tasks compared with our baseline and prior work.

1 Introduction

Information extraction (IE) has long been considered a fundamental challenge for various downstream natural language understanding tasks, such as knowledge graph construction and reading comprehension, *etc*. The goal is to identify and extract structured information from unstructured natural language text, such that both users and machines can easily comprehend the entities, relations, and events within the text.

Typically, IE consists of a series of different tasks to recognize entities, connect coreferences, extract relations, detect events, and so on. Conventional IE schemes commonly treat different IE tasks separately, while neglecting *cross-instance* (*e.g.*, event triggers, entities, roles, relations) or *cross-task* dependencies. Such isolated learning and inference schemes lead to severely insufficient knowledge capturing and inefficient model constructions. Intuitively, predictions of different IE instances from the same or different tasks can influence each other. For example, a relation between two entities would restrict the types of the entities (*e.g.*, two entities linked by a PART-WHOLE relation are more likely to share entity types of the same nature, as shown in the first example of Figure 1); types of entities can provide information that is useful to predict their relations or limit the roles they play in certain events (*e.g.*, the knowledge of event Life:Die and entity PER can benefit the prediction of the role Victim, as shown in the second example of Figure 1).

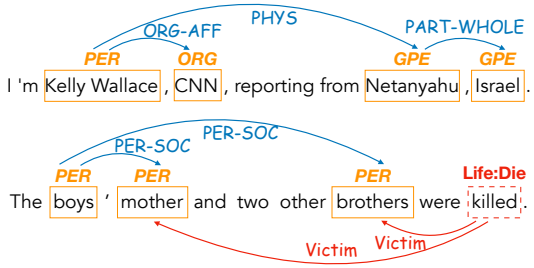


Figure 1: Example annotations of entity recognition (*e.g.*, PER), relation extraction (*e.g.*, PER-SOC) and event extraction tasks (*e.g.*, Life:Die and Victim).

To effectively capture instance or task dependencies, joint IE tries to simultaneously predict instances of different IE tasks for an input text with a multitask learning scheme, which attracts lots of interest and demonstrates significant improvements over specific-task learning methods. Previous work of joint IE focuses on three directions: 1) *representation enrichment* by sharing the token encoder between different IE tasks (Luan et al., 2018), up-

*This work was conducted when Zixia Jia was a research intern at BIGAI.
†Correspondence to Zilong Zheng and Kewei Tu.

Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 13695–13710

dating shared span representations according to local task-specific predictions (Luan et al., 2019a; Wadden et al., 2019), creating dependency graphs between instances (Lin et al., 2020; Zhang and Ji, 2021; Van Nguyen et al., 2021), or leveraging external dependency relations such as abstract meaning representation (AMR) and syntactic structures (Zhang and Ji, 2021; Van Nguyen et al., 2022a); 2) *type dependency scoring* by forming type patterns constraints (Lin et al., 2020), designing type dependency graphs (Van Nguyen et al., 2021), learning transition matrix of type pairs (Van Nguyen et al., 2022a), or computing mutual information (MI) scores of each pair of types (Van Nguyen et al., 2022b); 3) *global decoding* by beam search according to global features or AMR graphs (Lin et al., 2020; Zhang and Ji, 2021), or adopting global optimization algorithms such as simulated annealing (Van Nguyen et al., 2022a). Our interest lies in the second and third directions and we find two main limitations of prior works. The first one is that they only score binary dependencies of instance types (*i.e.* constraint, transition, or MI scores between a pair of types). The second one is that their decoders are based on discrete local search strategies to approximate global optima, and they often employ different approximate strategies for inference and training.

To alleviate aforementioned limitations, we propose a novel joint IE framework, Information Extraction as high-order CRF (CRFIE), that *explicitly* models label correlations between different instances from the same or different tasks, and utilizes them to calculate a joint distribution for final instance label predictions. Specifically, we demonstrate the effectiveness of our proposed high-order framework on three widely-explored IE tasks: entity recognition (EntR), relation extraction (RelE) and event extraction (EventE). We formulate the three tasks as a unified graph prediction problem, further modeled as a high-order Conditional Random field (CRF) (Ghamrawi and McCallum, 2005), where variables contain node variables and edge variables representing trigger/entity instances and role/relation instances respectively. The term "high-order" refers to factors connecting two or more correlated variables. Beyond the unary (first-order) factor, we design not only the binary (second-order) factor to model the interactions between a pair of edge variables but also the ternary (third-order) factor to model the interactions between node-edge-node variables. Since the correlated instances may come from the same or different tasks, we categorize our high-order factors into two types: **homogeneous factors (homo)** representing correlations between instances of the same task, and **heterogeneous factors (hete)** representing correlations between instances of different tasks. Taking EntR and EventE as an example, we calculate binary factor potentials of role-role pairs (homo), and ternary factor potentials of trigger-role-entity triplets (hete). We leverage these scores to predict the labels of all instances jointly. Since exact high-order inference is analytically intractable, we incorporate a neural decoder that is unfolded from the approximate Mean-Field Variational Inference (MFVI) (Xing et al., 2012) method, which achieves end-to-end training and also consistent inference and learning processes. Note that MFVI can be seen as a continuous relaxation for CRF inference (Lê-Huu and Alahari, 2021), which can often be more effective than discrete optimization used in previous work. Experiments on joint IE tasks show that CRFIE achieves competitive or better performance compared with previous state-of-the-art models[1].

2 Method

2.1 Overview of Joint IE as Graph Prediction

We investigate three widely-explored IE tasks.

▷ EntR aims to identify some spans in a sentence as entities and label their entity types.

▷ RelE aims to identify relations between some entity pairs and label their relation types.

▷ EventE aims to label event types and its trigger words, identify some entities as event arguments and label argument roles.

We formulate the three IE tasks as a graph $G = (V, E)$ prediction task, where V denotes the node set and E denotes the directed edge set. Each node $v = (a, b, l) \in V$ is a span for a trigger or an entity, where a and b index the start and end words of the span, and $l \in \mathcal{L}^{\text{event}}$ or $l \in \mathcal{L}^{\text{entity}}$ denotes the node's event type or entity type, respectively. Each edge $e_{ij} = (i, j, r) \in E$ represents the relationship from node v_i to node v_j, and $r \in \mathcal{R}^{\text{role}}$ or $r \in \mathcal{R}^{\text{relation}}$ represents the edge label which is a role type when the edge is from a trigger to an entity (as an argument) or a relation type when the edge is from one entity to another.

[1]The code can be found at https://github.com/JZXXX/High-order-IE.

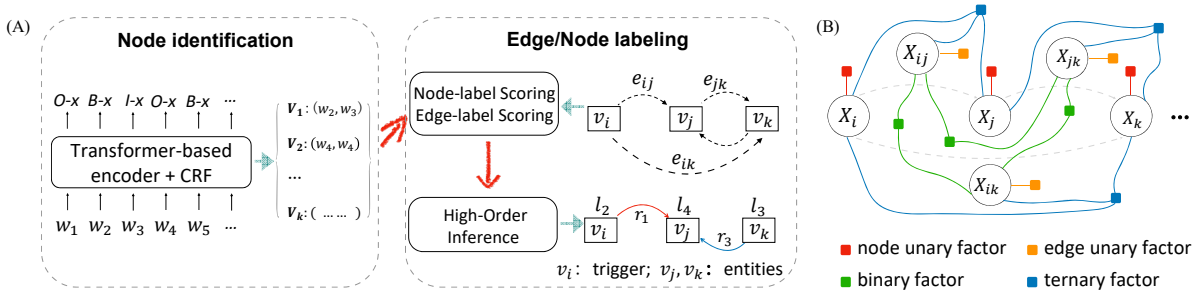


Figure 2: **An overview of CRFIE.** (A) Model architecture. The identification module provides spans as nodes to the node/edge labeling module. (B) An example factor graph of our node/edge labeling module containing variables representing three nodes and three edges. X_i indicates the label variable of the i-th node v_i and X_{ij} indicates the label variable of the edge e_{ij} from the v_i to v_j. The node labels can be event types or entity types (*i.e.*, X_i is the abbreviation of X_i^{ntask} for simplicity and $ntask \in \{\text{event, entity}\}$). The edge labels can be relations or argument roles (*i.e.*, X_{ij} is the abbreviation of X_{ij}^{etask} and $etask \in \{\text{relation, role}\}$). For simpler illustration, we omit edges of the opposite direction.

Figure 2(A) depicts the overall architecture of CRFIE. Because joint identification and classification need to enumerate all possible spans as nodes and high-order inference whose complexity is related to the node number becomes too computationally expensive in this situation, we follow previous work (Lin et al., 2020; Zhang and Ji, 2021; Van Nguyen et al., 2021, 2022a) and adopt the following pipeline: first extracting graph nodes with a node identification module, and then predicting labels of nodes and edges with a node/edge labeling module.

The **node identification module** aims to identify spans in the input sentence as graph nodes. This module is not the focus of our work, so we simply follow previous work (Lample et al., 2016a; Lin et al., 2020; Zhang and Ji, 2021; Van Nguyen et al., 2021) to formulate node identification as a sequence labeling task with a BIO scheme. Specifically, after getting word features by averaging all sub-word embeddings extracted from a pre-trained transformer-based encoder, such as BERT (Devlin et al., 2018), we use two vanilla linear-chain conditional random field (CRF) (Lafferty et al., 2001) as decoders to acquire trigger nodes and entity nodes separately. We follow the conventional joint IE settings without considering nested spans. More advanced methods such as Yu et al. (2020); Lou et al. (2022) can be adopted to identify graph nodes if span nesting needs to be considered. More details about the identification module can be found in Appendix A. The identification module is fixed during subsequent training of the node/edge labeling module.

The **node/edge labeling module** is designed to predict (i) an event type for each trigger node and an entity type for each entity node and (ii) a role type for each edge between a trigger-entity pair and a relation type for each edge between an entity-entity pair. We use a special NULL label to represent non-existence of an edge. We formulate the node/edge labeling module as a high-order CRF, illustrated as a factor graph in Figure 2(B). There are three kinds of factors: *unary factors* that reflect the likelihood of each variable's label; *binary factors* for pairs of edges sharing an endpoint, which models correlations between edge variables; and *ternary factors* for an edge, its head node and its tail node, which models correlations between related node and edge variables. The joint probability over all the variables is proportional to the exponentiated sum of all the score function values of such factors. Due to the intractability of exact high-order inference, we use MFVI to approximate it. A multitask learning scheme is adopted to train our node/edge labeling module. We describe the scoring functions, high-order inference, and learning method in the following subsections in detail.

2.2 Unary Scoring

We first obtain each node's representation $\mathbf{z}$ by averaging the representations of all the words within a span, in which the words' representations are obtained in the same way as in the identification module, but from another pre-trained transformer-based encoder. Then, the unary scores of the i-th node labels $\mathbf{s}_i^{\text{u-}ntask} \in \mathbb{R}^{|\mathcal{L}^{ntask}|}$ can be obtained by feeding $\mathbf{z}_i$ into a two layers task-specific feed-forward

neural network (FNN):

$$\mathbf{s}_i^{\text{u-}ntask} = \text{FNN}^{ntask}(\mathbf{z}_i), \quad (1)$$

where $\mathcal{L}^{ntask}$ represents a task-specific node label set, and $ntask \in \{\text{event, entity}\}$.

The unary scores $\mathbf{s}_{ij}^{\text{u-}etask}$ of an edge e_{ij} from v_i to v_j can be computed with a decomposed biaffine function:

$$\mathbf{s}_{ij}^{\text{u-}etask} = (\text{FNN}^{etask\text{-s}}(\mathbf{z}_i) \circ \text{FNN}^{etask\text{-e}}(\mathbf{z}_j))\mathbf{H}^{\text{u-}etask}$$

where two task-specific FNNs are single-layer, $\mathbf{H}^{\text{u-}etask} \in \mathbb{R}^{d_{etask} \times |\mathcal{R}^{etask}|}$ is parameters, $\mathcal{R}^{etask}$ represents a task-specific edge label set that includes an additional NULL label, $etask \in \{\text{relation, role}\}$, and $\circ$ denotes element-wise product.

2.3 Binary Scoring

We calculate binary correlation scores of each legal edge pair that share one endpoint. As illustrated in Figure 2(A), there are three types of binary factors (Wang et al., 2019b): edge e_{ij} and edge e_{ik} share the head node v_i, producing sibling (sib); edge e_{jk} and edge e_{ik} share the tail node v_k, producing co-parent (cop); and the tail node v_j of edge e_{ij} is the head node of edge e_{jk}, producing grandparent (gp). For each specific type of binary factor, we use different single-layer FNNs taking $\mathbf{z}$ as input to calculate a head representation (-s) and a tail representation (-e) for each node. For gp factor, we additionally calculate a middle representation (-mid) for each node.

$$\mathbf{g}_i^{type\text{-s}} = \text{FNN}^{type\text{-s}}(\mathbf{z}_i) \quad \mathbf{g}_i^{type\text{-e}} = \text{FNN}^{type\text{-e}}(\mathbf{z}_i)$$
$$\mathbf{g}_i^{\text{gp-mid}} = \text{FNN}^{\text{gp-mid}}(\mathbf{z}_i) \quad type \in \{\text{sib, cop, gp}\}$$

For a sib pair $\{e_{ij}, e_{ik}\}$, cop pair $\{e_{ik}, e_{jk}\}$ and gp pair $\{e_{ij}, e_{jk}\}$, suppose that the first edge has label $r_m \in \mathcal{R}^1$ and the second edge has label $r_n \in \mathcal{R}^2$, we formulate binary scores as follows:

$$s_{ijkmn}^{\text{b-sib}} = \sum_{a=1}^{d_3} (\mathbf{g}_i^{\text{sib-s}} \circ \mathbf{g}_j^{\text{sib-e}} \circ \mathbf{g}_k^{\text{sib-e}} \circ \mathbf{h}_m^1 \circ \mathbf{h}_n^2)_a$$
$$s_{ijkmn}^{\text{b-cop}} = \sum_{a=1}^{d_3} (\mathbf{g}_i^{\text{cop-s}} \circ \mathbf{g}_j^{\text{cop-s}} \circ \mathbf{g}_k^{\text{cop-e}} \circ \mathbf{h}_m^1 \circ \mathbf{h}_n^2)_a$$
$$s_{ijkmn}^{\text{b-gp}} = \sum_{a=1}^{d_3} (\mathbf{g}_i^{\text{gp-s}} \circ \mathbf{g}_j^{\text{gp-mid}} \circ \mathbf{g}_k^{\text{gp-e}} \circ \mathbf{h}_m^1 \circ \mathbf{h}_n^2)_a$$

where $\mathbf{h}_m^1$ is the embedding of the first edge label r_m and $\mathbf{h}_n^2$ is the embedding of the second edge label r_n. All $\mathbf{g}$ and $\mathbf{h}$ are d_3-dimensional. For symmetry, $s_{ijkmn}^{\text{b-sib}} \equiv s_{ikjnm}^{\text{b-sib}}$ and $s_{ijkmn}^{\text{b-cop}} \equiv s_{jikmn}^{\text{b-cop}}$.

In this paper, we consider two types of homogeneous binary factors: ***homo* case (i)** sib and cop representing two argument roles ($\mathcal{R}^1 = \mathcal{R}^2 = \mathcal{R}^{\text{role}}$) and ***homo* case (ii)** sib, cop and gp representing two relations ($\mathcal{R}^1 = \mathcal{R}^2 = \mathcal{R}^{\text{relation}}$). We also consider one type of heterogeneous binary factors: ***hete* case (i)** cop and gp where one edge label is a relation and the other is a role for joint EventE and RelE ($\mathcal{R}^1 = \mathcal{R}^{\text{relation}}, \mathcal{R}^2 = \mathcal{R}^{\text{role}}$ or $\mathcal{R}^1 = \mathcal{R}^{\text{role}}, \mathcal{R}^2 = \mathcal{R}^{\text{relation}}$).[2]

2.4 Ternary Scoring

We calculate ternary correlation scores of an edge and its two endpoints. Similar to binary scoring, we use two new FNNs to produce representations for each possible head node and tail node respectively:

$$\mathbf{g}_i^{\text{ter-s}} = \text{FNN}^{\text{ter-s}}(\mathbf{z}_i) \qquad \mathbf{g}_i^{\text{ter-e}} = \text{FNN}^{\text{ter-e}}(\mathbf{z}_i)$$

For an edge with label $r_m \in \mathcal{R}$, its head node v_i having label $l_p \in \mathcal{L}^s$ and its tail node v_j having label $l_q \in \mathcal{L}^e$, the ternary score is calculated as:

$$s_{ijpqm}^{\text{ter}} = \sum_{a=1}^{d_4} (\mathbf{g}_i^{\text{ter-s}} \circ \mathbf{g}_j^{\text{ter-e}} \circ \mathbf{e}_p^{\text{ter-s}} \circ \mathbf{e}_q^{\text{ter-e}} \circ \mathbf{h}_m^{\text{ter}})_a \quad (2)$$

where $\mathbf{h}_m^{\text{ter}}$ is the embedding of label r_m, $\mathbf{e}_p^{\text{ter-s}}$ is the embedding of label l_p and $\mathbf{e}_q^{\text{ter-e}}$ is the embedding of label l_q. $\mathbf{g}$, $\mathbf{e}$ and $\mathbf{h}$ are all d_4-dimensional. We consider two types of heterogeneous ternary factors: ***hete* case (ii)** the ternary correlations between an event trigger, an entity, and a role for joint EventE and EntR ($\mathcal{L}^s = \mathcal{L}^{\text{event}}$, $\mathcal{R} = \mathcal{R}^{\text{role}}$ and $\mathcal{L}^e = \mathcal{L}^{\text{entity}}$) and ***hete* case (iii)** two entities and their relation for joint RelE and EntR ($\mathcal{L}^s = \mathcal{L}^e = \mathcal{L}^{\text{entity}}$ and $\mathcal{R} = \mathcal{R}^{\text{relation}}$).

2.5 High-Order Inference

In contrast to first-order inference which independently predicts the value of each variable by maximizing its unary score, in high-order inference we jointly predict the values of all the variables to maximize the sum of their unary and high-order scores. However, the exact joint inference on our factor graph is NP-hard in general. Therefore, we use Mean-Field Variational Inference (MFVI) (Xing et al., 2012) for approximate inference. MFVI iteratively updates an approximate posterior marginal distribution $Q(X)$ of each variable X based on

[2] It is rare that a trigger word serves as an argument meanwhile, and a relation edge and a role edge scarcely share the same head node, so we do not consider gp in ***homo* case (i)** and sib in ***hete* case (i)**.

messages from all the factors connected to it. For simplicity, we write $Q_i(l)$ and $Q_{ij}(r)$ to denote $Q(X_i = l)$ and $Q(X_{ij} = r)$ respectively.

Messages for edge variables aggregated from binary factors are calculated as:

$$F_{\text{bi}}^{(t)}(X_{ij} = r_m) = \sum_{k \neq i,j} \sum_{r_n \in \mathcal{R}^2} \alpha_1 s_{ijkmn}^{\text{sib}} Q_{ik}^{(t)}(r_n) + \alpha_2 s_{ikjmn}^{\text{cop}} Q_{kj}^{(t)}(r_n) + \alpha_3 \big(s_{ijkmn}^{\text{gp}} Q_{jk}^{(t)}(r_n) + s_{kijmn}^{\text{gp}} Q_{ki}^{(t)}(r_n) \big)$$

where $\alpha_1, \alpha_2, \alpha_3 \in [0, 1]$ are hyper-parameters controlling the scale of messages passed by the different types of binary factors. These hyper-parameters are not part of standard MFVI and can instead be seen as part of the scoring function.

Messages for node variables and edge variables aggregated from ternary factors are calculated as:

$$F_{\text{ter}}^{(t)}(X_{ij} = r_m) = \sum_{l_p \in \mathcal{L}^s} \sum_{l_q \in \mathcal{L}^e} s_{ijpqm}^{\text{ter}} Q_i^{(t)}(l_p) Q_j^{(t)}(l_q)$$

$$F_{\text{ter}}^{(t)}(X_i = l_p) = \sum_{l_q \in \mathcal{L}^e} \sum_{r_m \in \mathcal{R}} s_{ijpqm}^{\text{ter}} Q_j^{(t)}(l_q) Q_{ij}^{(t)}(r_m)$$

$$F_{\text{ter}}^{(t)}(X_j = l_q) = \sum_{l_p \in \mathcal{L}^s} \sum_{r_m \in \mathcal{R}} s_{ijpqm}^{\text{ter}} Q_i^{(t)}(l_p) Q_{ij}^{(t)}(r_m)$$

The posterior $Q(X)$ is updated based on the messages as follows:

$$Q_{ij}^{(t+1)}(r_m) \propto \exp\{ s_{ijm}^{\text{u-}etask} + \alpha_4 F_{\text{bi}}^{(t)}(X_{ij} = r_m) + \alpha_5 F_{\text{ter}}^{(t)}(X_{ij} = r_m) \}$$

$$Q_i^{(t+1)}(l_p) \propto \exp\{ s_{ip}^{\text{u-}ntask} + \alpha_6 F_{\text{ter}}^{(t)}(X_i = l_p) \}$$

$$Q_j^{(t+1)}(l_q) \propto \exp\{ s_{jq}^{\text{u-}ntask} + \alpha_7 F_{\text{ter}}^{(t)}(X_j = l_q) \}$$

where all $\alpha \in [0, 1]$ are hyper-parameters controlling the scale of different types of messages, $s_{ijm}^{\text{u-}etask}$ is the m-th element of the unary potential $\mathbf{s}_{ij}^{\text{u-}etask}$, $s_{ip}^{\text{u-}ntask}$ is the p-th element of the unary potential $\mathbf{s}_{i}^{\text{u-}ntask}$ and $s_{jq}^{\text{u-}ntask}$ is the q-th element of $\mathbf{s}_{j}^{\text{u-}ntask}$.

There are two ways of iterative MFVI update. In the synchronous update, we update $Q(X)$ for all the variables at each step. In asynchronous update, we alternate between node variables and edge variables for $Q(X)$ update. We empirically find that asynchronous update is better than synchronous update when we use ternary factors in some cases.

The initial distribution $Q^{(0)}$ is set by normalizing exponentiated unary potentials. After a fixed T (which is a hyper-parameter) number of iterations, we obtain the posterior distribution $Q^{(T)}$. For each variable, we pick the label with the highest probability according to $Q^{(T)}$ as our prediction.

2.6 Multitask Learning

Given a sentence $\mathbf{w} = (w_1, ..., w_k)$, to train multiple IE tasks with our unified high-order node-relation prediction framework, we do multi-task learning with cross-entropy losses as follows:

$$\mathcal{L} = -\sum_i \log P(\hat{X}_i^{ntask} | \mathbf{w}) - \sum_{i,j} \log P(\hat{X}_{ij}^{etask} | \mathbf{w})$$

where $\hat{X}_i^{ntask}$ and $\hat{X}_{ij}^{etask}$ denote the ground truth labels of nodes and edges respectively for all the tasks. The conditional distributions over node labels and edge labels with first-order inference are

$$P(X_i^{ntask} | \mathbf{w}) = (\text{SoftMax}(\mathbf{s}_i^{\text{u-}ntask}))_{X_i^{ntask}}$$

$$P(X_{ij}^{etask} | \mathbf{w}) = (\text{SoftMax}(\mathbf{s}_{ij}^{\text{u-}etask}))_{X_{ij}^{etask}}$$

and those with high-order inference are:

$$P(X_i^{ntask} | \mathbf{w}) = Q_i^{(T)}(X_i^{ntask})$$

$$P(X_{ij}^{etask} | \mathbf{w}) = Q_{ij}^{(T)}(X_{ij}^{etask}).$$

where $Q^{(T)}$ is computed with T MFVI iterations.

Inspired by Zheng et al. (2015); Wang et al. (2019b), we unfold the MFVI iteration steps as recurrent neural network layers parameterized by unary and high-order scores. As such, we obtain an end-to-end recurrent neural network for both inference and training. Doing this has an added benefit of consistent inference and training, unlike traditional CRF approaches that may rely on different approximation methods for inference and training (see for example Van Nguyen et al. (2022a)).

3 Experiments

Datasets We evaluate our model on the ACE2005 corpus (Walker et al., 2005) which provides entity, relation, and event annotations. Following Lu et al. (2021); Lin et al. (2020); Wadden et al. (2019), we conduct experiments on four English datasets: ACE05-R for EntR and RelE, ACE05-E for EntR and EventE, and ACE05-E+ and ERE-EN for all the three tasks, with the same data

pre-processing and train/dev/test split. There are 7 entity types, 6 relation types, 33 event types, and 22 argument roles defined in the ACE2005 corpus. ERE-EN dataset is extracted by combining the data from three datasets for English (i.e., LDC2015E29, LDC2015E68, and LDC2015E78) that are created under Deep Exploration and Filtering of Test (DEFT) program. It includes 7 entity types, 5 relation types, 38 event types, and 20 argument roles. Statistics of all datasets we used are shown in Tabel 1.

	Split	#Sents	#Entities	#Relations	#Events
ACE05-R	Train	10,051	26,473	4,788	-
	Dev	2,424	6,362	1,131	-
	Test	2,050	5,476	1,151	-
ACE05-E	Train	17,172	29,006	4,664	4,202
	Dev	923	2,451	560	450
	Test	832	3,017	636	403
ACE05-E+	Train	19,216	47,525	7,152	4,419
	Dev	902	3,422	728	468
	Test	676	3,673	802	424
ACE05-CN	Train	6841	29657	7934	2926
	Dev	526	2250	596	217
	Test	547	2388	672	190
ERE-EN	Train	14736	39501	5054	6208
	Dev	1209	3369	408	525
	Test	1163	3295	466	551

Table 1: Datasets statistics

Evaluation We use F1 scores to evaluate our model's performance as in most previous work (Lu et al., 2021; Lin et al., 2020; Wadden et al., 2019; Zhang and Ji, 2021). For the EntR task, an entity (*Ent*) is correct if both its type and offsets match a gold entity. For the RelE task, a relation (*Rel*) is correct if both its type and the offsets of its two related entities match a gold relation. In addition, a strict relation evaluation (*Rel+*) requires that the types of the two related entities are also correct. A trigger is correctly identified (*Trig-I*) if its offsets match a gold trigger. It is correctly classified (*Trig-C*) if its corresponding event type also matches the reference trigger. An argument is correctly identified (*Arg-I*) if its offsets match a gold argument and its corresponding event type is correct. It is correctly classified (*Arg-C*) if its role type also matches the reference argument. All experimental results of our approach shown in this paper are the average of three runs with different random seeds.

Implementation Details For fair comparison with previous state-of-the-art systems, we use the BERT-large-cased model (Devlin et al., 2018) or RoBERTa model (Liu et al., 2019) as our encoder for the ACE05-E and ACE05-E+ datasets, and ALBERT model (Lan et al., 2019) as the encoder for the ACE05-R dataset. We train our model with BertAdam optimizer[3]. When we use a single kind of factor, α is set to 1 for the used and set to 0 for others. When multiple kinds of factors are used, α of the used are tunable parameters. Detailed hyperparameter values are provided in Appendix B.

	Ent	*Tri-I*	*Tri-C*	*Arg-I*	*Arg-C*
DYGIE++ (Wadden et al., 2019)†	89.7	-	69.7	53.0	48.8
Zhang et al. (2019)◦	87.1	73.9	72.0	57.2	52.4
OneIE (Lin et al., 2020)†	90.2	78.2	74.7	59.2	56.8
Text2Event (Lu et al., 2021)*	-	-	71.9	-	53.8
FourIE (Van Nguyen et al., 2021)†	91.3	78.3	75.4	60.7	58
FourIE (Van Nguyen et al., 2021)‡	91.6	-	74.9	-	58.7
CRFIE baseline†	90.8	77.7	74.8	58.5	56.4
CRFIE *homo* case (i)†	90.8	77.7	74.6	58.7	57.1
CRFIE *hete* case (ii)†	90.7	77.7	74.3	59.2	57.2
CRFIE *homo* case (i) + *hete* case (ii)†	90.6	77.7	74.3	59.6	57.5
CRFIE baseline‡	91.5	77.2	73.6	60.8	58.1
CRFIE *homo* case (i)‡	91.4	77.2	73.5	61.3	58.8
CRFIE *hete* case (ii)‡	91.7	77.2	73.7	61.9	59.4
CRFIE *homo* case (i) + *hete* case (ii)‡	91.5	77.2	73.8	61.9	59.1
FOR REFERENCE					
AMRIE (Zhang and Ji, 2021)‡	92.1	78.1	75	60.9	58.6
GraphIE (Van Nguyen et al., 2022a)‡	91.4	-	75.1	-	59.4

Table 2: Average F1 on ACE05-E dataset. ◦, ∗, †, ‡ mean ELMo, T5-large, BERT-large-cased and RoBERTa-large encoder, respectively. The results of FourIE (RoBERTa) are from Van Nguyen et al. (2022a). The results of AMRIE and GraphIE are listed for reference because they use external resources (AMR graph and syntactic tree).

3.1 Main Results

We take our framework with first-order inference (i.e., independently predicting the value of each variable by maximizing its unary score) as **CRFIE baseline**. It can be seen that our baseline performs better than previous work in some cases, which benefits from the biaffine function in calculating unary scores. We experiment with different combinations of tasks.

Joint EntR, EventE We compare our approach under different settings and also with previous work that did not leverage gold triggers and entities. Table 2 shows the experimental results. The cases in the table (*e.g.*, ***homo* case (i)**) are corresponding to the aforementioned settings in the subsections 2.3 and 2.4. The F1 scores of *Tri-I* of different settings are the same because they are produced by the same node identification module that is fixed to fairly compare our model in different settings.

[3]https://github.com/huggingface/transformers

	Ent	*Rel*	*Rel+*
DYGIE++ (Wadden et al., 2019)†	88.6	63.4	-
OneIE (Lin et al., 2020)†	88.8	67.5	
Wang and Lu (2020)Δ	89.5	67.6	64.3
PURE$_s$ (Zhong and Chen, 2020)Δ	89.7	69.0	65.6
UNIRE (Wang et al., 2021)Δ	90.2	-	66.0
PFN (Yan et al., 2021)Δ	89.0	-	66.8
FourIE (Van Nguyen et al., 2021)†	88.9	68.9	-
UIE (Lu et al., 2022)*	-	-	66.1
CRFIE baselineΔ	89.8	69.9	67.5
CRFIE *homo* case (ii)Δ	90.2	70.8	68.2
CRFIE *hete* case (iii)Δ	90.1	70.4	68.3
FOR REFERENCE			
GraphIE (Van Nguyen et al., 2022a)‡	89.3	68.5	-
PURE$_c$ (Zhong and Chen, 2020)Δ	90.9	69.4	67.0
PL-Marker$_{\text{re-eval}}$ (Ye et al., 2022)Δ	91.3	72.5	70.5

Table 3: Average F1 on ACE05-R dataset. Subscript of *re-eval* means re-evaluation (Appendix F) using the standard evaluation method as other work. $*$, †, ‡ and Δ mean T5-large, BERT-large-cased, RoBERTa-large and ALBERT-XXLarg-v1, respectively. PURE$_s$ refers to the PURE model with single-sentence features. The results of PURE$_c$ and PL-Marker are listed for reference because they use cross-sentence features and are not directly comparable with other models. The reason for GraphIE listed for reference is the same as in Tabel 2.

ACE05-E+	*Ent*	*Rel*	*Tri-I*	*Tri-C*	*Arg-I*	*Arg-C*
OneIE Lin et al. (2020)	89.6	58.6	75.6	72.8	57.3	54.8
Text2Event Lu et al. (2021)*	-	-	-	71.8	-	54.4
FourIE (Van Nguyen et al., 2021)	91.1	63.6	76.7	73.3	59.5	57.5
UIE Lu et al. (2022)*	-	-	-	73.4	-	54.8
GTEE-DYNPREF Liu et al. (2022)	-	-	-	74.3	-	54.7
CRFIE baseline	90.8	65.3	77.4	74.6	60.0	58.1
CRFIE *hete* case (i)	90.7	65.1	77.4	74.8	60.3	58.5
CRFIE all	90.9	65.8	77.4	75.5	60.8	58.8
FOR REFERENCE						
GraphIE (Van Nguyen et al., 2022a)	91.0	65.4	-	74.8	-	59.9

ERE-EN	*Ent*	*Rel*	*Tri-I*	*Tri-C*	*Arg-I*	*Arg-C*
OneIE Lin et al. (2020)	86.3	52.8	66.0	57.1	43.7	42.1
CRFIE baseline‡	87.6	54.4	69.9	61.5	45.9	44.2
CRFIE all‡	87.4	55.1	69.9	61.4	53.5	51.2
FOR REFERENCE						
AMRIE (Zhang and Ji, 2021)‡	87.9	55.2	68	61.4	46.4	45.0

Table 4: Average F1 on ACE05-E+ and ERE-EN datasets. $*$ means T5-large, ‡ means RoBERTa-large. Others without mark use BERT-large-cased. The reason for reference is the same as in Table 2. We do not compare FourIE and GraphIE on ERE-EN dataset because their splittings of train/dev/test are different from ours. The results of previous work on ERE-EN are from Zhang and Ji (2021).

It can be seen that our high-order model performs better than our baseline in most cases for EventE, which directly shows the benefit of high-order factors. Compared to previous SOTA, our model performs uncompetitive on *Tri-I*, because we focus on the interactions of node/edge labeling, and we did not tune the hyper-parameters of the node identification module while just keeping them the same as Lin et al. (2020). Even with an unsatisfactory identification module, the results of *Arg-C* which is the most difficult sub-task in EventE show that CRFIE achieves consistent improvement. It is worth noting that CRFIE with learned dependencies can achieve comparable performance with those models (Zhang and Ji, 2021; Van Nguyen et al., 2022a) leveraging external syntactic or semantic dependencies. It is surprising that when we use both binary factors (***homo* case (i)**)and ternary factors (***hete* case (ii)**) in the RoBERTa setting, the performance slightly drops. The reason may be that messages from different types of factors may conflict with each other, such that training becomes more difficult. We also experiment in the case where gold triggers and entities are given, results are shown in Appendix C.

Joint EntR and RelE Table 3 shows our experimental results on the ACE05-R dataset. We can find that CRFIE performs better than most previous work and our baseline both on EntR and RelE, which demonstrates the advantage of high-order inference. Similar to joint EntR and EventE, our high-order model with the combination of all factors cannot achieve further improvement, so we do not show the result of this setting.

Joint EntR, EventE and RelE Table 4 shows the experimental results on the ACE05-E+ and ERE-EN datasets. On ACE05-E+, we show the result of ***hete* case (i)** because this setting is not included in the above experiments. **CRFIE all** means that we use all kinds of binary and ternary factors that have performed benefits in ablation experiments. We can find that CRFIE achieves consistent improvement in EventE and RelE. Due to the space limitation, more ablations and experimental results can be found in Appendix D.

3.2 Analysis

High-Order Scoring We study two variants of our high-order scoring. *Share* means that we reuse the label representations in unary scoring for high-order scoring instead of using new label representations. *W/o node reps* means that we calculate high-order scores without taking node representations into account, such that the high-order scores are only dependent on the labels regardless of the underlying text spans that constituent the nodes and edges. Table 5 shows the comparison results with ternary factors on the ACE05-R dataset. We can find that the performance of the two variants both drops.

	Ent	*Rel*	*Rel+*
Ours *hete* (+ter)	90.1	70.4	68.3
Share	90.0	69.7	67.5
W/o node reps	90.1	70.0	67.7

Table 5: Comparison of the results of different high-order scoring methods on ACE05-R dataset.

	Ent	*Tri-I*	*Tri-C*	*Arg-I*	*Arg-C*
Asyn (BERT)	90.9	77.7	74.3	59.2	57.2
Syn (BERT)	90.7	77.7	74.8	59.2	56.9
Asyn (RoBERTa)	91.7	77.2	73.7	61.9	59.4
Syn (RoBERTa)	91.7	77.2	73.7	61.3	58.8

Table 6: Comparison of the results of synchronous and asynchronous updating strategies when we use ternary factor on ACE05-E dataset.

	baseline	*+sib*	*+ter*	*+sib+ter*
Train	119.3	119.2	118.4	107.6
Test	91.2	85.1	81.4	77.2

Table 7: Comparisons of speed (sentences/second) among the baseline and high-order models.

Message Passing of Ternary Factors From the message passing process involving ternary factors in Sec. 2.5, we can see that messages passed to an edge come only from its two endpoints, but a node gets messages from all possible edges connected to it, which causes asymmetry messages from ternary factors, we try synchronous and asynchronous updating strategies as described in Sec 2.5. For asynchronous updating, we firstly update edge posteriors using node posteriors for the reason that the initial node posteriors are more accurate. Table 6 shows the comparison results of the two updating strategies on the ACE05-E dataset. We can find that asynchronous update has an advantage over synchronous update on *Arg-C* but harms or keeps the performance on *Tri-C*.

Complexity and Speed of High-order Inference The computational complexity of our high-order inference is $O(n^3|\mathcal{R}|^2 + n|\mathcal{L}|)$ when we consider binary factors and $O(n^2|\mathcal{R}||\mathcal{L}|^2)$ when we consider ternary factors, while our first-order model has a computational complexity of $O(n^2|\mathcal{R}| + n|\mathcal{L}|)$, where n is the node number. We measure the empirical training speed and inference speed on an A100 server (Table 7). We can find that our high-order models are only slightly slower than the baseline despite the difference in computational complexity, which is because we implement our models with full GPU parallelization.

Visualization of Correlation Score We take relation extraction as an example to visualize the ternary score calculated by Eq. 2 between entity-relation-entity triplets. For better understanding, we show examples of selected entity types and relation types. From Fig. 4, we can find that the correlation scores can reflect some prior knowledge. For example, 'PER-SOC' relation exists between two 'PER' entities, 'PART-WHOLE' relation is more likely to exist between entities with the same types.

Error Correction Analysis and Case Study We provide quantitative error correction analysis in Appendix E. Figure 3 shows examples where our high-order approach revises wrong predictions made based on the initial unary scores (i.e., the first-order baseline), along with our analyses of how high-order factors achieve the revision.

4 Related Work

Information Extraction Classical IE models are typically task-specific (Lample et al., 2016b; Yu et al., 2020; Zeng et al., 2014; Wang et al., 2019a). Recent efforts develop joint methods for multiple IE tasks (Miwa and Sasaki, 2014; Zheng et al., 2017; Nguyen and Nguyen, 2019; Zhang et al., 2019; Wang and Lu, 2020) or general architectures for universal IE (Paolini et al., 2021; Lu et al., 2022; Lou et al., 2023). Graph-based joint IE methods formulate multiple IE tasks as a graph prediction task and aim to capture dependencies between different instances or tasks. Lots of previous works leverage encoder sharing or graph convolutional networks (GCNs) on instance dependency graphs to enrich instance representations (Wadden et al., 2019; Fu et al., 2019; Van Nguyen et al., 2021, 2022a,b). This work is more relevant to some recent works that take efforts on type interactions and global inference. Lin et al. (2020) manually designs global features as constraints and leverages beam search to find approximated global optima. Based on the method of Lin et al. (2020), Van Nguyen et al. (2021) further incorporates AMR graphs as external dependencies. The work of Van Nguyen et al. (2022a) is more similar to ours in that they adopt a CRF to model type dependencies, but they learn a transition matrix that only scores binary dependencies. Besides, they employ Noise Contrastive Estimation (NCE) (Mikolov et al., 2013) to perform approximate training and Simulated An-

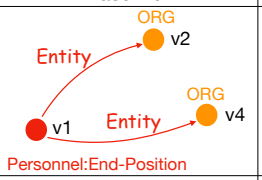
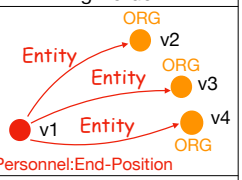
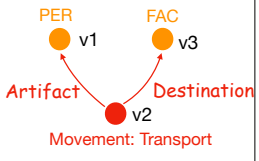
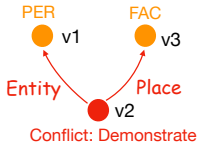
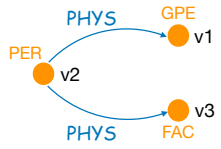
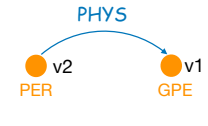
Sentence & Analysis	Baseline	High-order
#1: As well as previously **(v1)** holding senior positions at Barclays Bank **(v2)**, BZW **(v3)** and Kleinwort Benson **(v4)**, McCarthy was formerly a top civil servant at the Department of Trade and Industry. Analysis: Sibling factor helps our high-order model find the *BZW* which is tied for *Barclays Bank* to be an argument of event *Personnel:End-Position* triggered by word *previously*.		
#2: The crowd **(v1)** filled **(v2)** the street **(v3)** leading to the Kazimiya mosque in the northeast of Baghdad and carried banners in the green color of Islam, calling for good government. Analysis: An entity with *PER* type has less possibility to play an *Artifact* role. Ternary factor leverages messages passed by node label distributions to refine the edge label which in turn gives the message to refine node labels.		
#3: For the most part the marches went off peacefully, but in New York **(v1)** a small group **(v2)** of protesters were arrested after they refused to go home **(v3)** at the end of their rally, police sources said. Analysis: There is less possibility that a *PER* entity has *PHYS* relation with *GPE* entity and *FAC* entity at the same time. Sibling and ternary factors help our high-order model in this situation.		

Figure 3: Examples showing how our high-order approach improves the graph prediction using different high-order factors. We only display a partial information graph for clearer illustration.

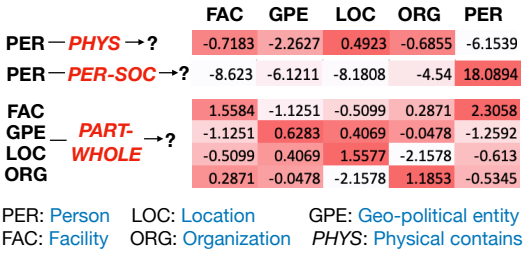


Figure 4: Ternary scores between entity-relation-entity triplets.

nealing Search to perform approximate inference. Different from their work, we model both binary and ternary dependencies and leverage MFVI to achieve consistent training and inference.

High-order Methods Previous high-order methods most focus on instance interactions in training process to get more expressive representations, such as sharing representations (Sun et al., 2019; Luan et al., 2019b) or using sequence-to-sequence architecture (Ma et al., 2022; Paolini et al., 2021; Lu et al., 2021). There are some high-order inference methods that are related to us on different NLP tasks. On dependency parsing, Wang and Tu (2020) considered three types of second-order parts of semantic dependencies and approximate decoding with mean-field variational inference or loopy belief propagation. Jia et al. (2022) considered interactions between two arguments of the same predicate on semantic role labeling task. However, due to the complexity, they only did high-order inference on edge existence prediction while leaving label prediction in first-order, and they did not involve heterogeneous factors. In another line of research, Wang and Pan (2020, 2021) integrate logic rules and neural network to leverage prior knowledge to help relation extraction and event extraction tasks. But they cannot achieve end-to-end training and inference.

5 Conclusion

In this paper, we propose a novel framework that leverages high-order interactions across different instances and different IE tasks in both training and inference processes. We formulate IE tasks as a unified graph prediction problem, further modeled as a high-order CRF. Our framework consists of an identification module to identify spans as graph nodes and a node/edge labeling module with high-order modeling and inference to jointly label all nodes and edges.

Limitations

The limitation is that we separate node identification and node/edge labeling processes. Because joint node identification and label classification should enumerate all possible spans in a sentence, which is too computationally expensive. Most previous works also separate the two processes. But an obvious disadvantage of such a pipeline scheme is the error propagation problem. We take joint node identification and label classification with

high-order inference as future work.

Acknowledgements

This work is supported in part by National Key R&D Program of China (2021ZD0150200) and the National Natural Science Foundation of China (61976139). Wenjuan Han is supported by the Talent Fund of Beijing Jiaotong University (2023XKRC006).

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Rakesh Dugad and UDAY B Desai. 1996. A tutorial on hidden markov models. *Signal Processing and Artificial Neural Networks Laboratory, Dept of Electrical Engineering, Indian Institute of Technology, Bombay Technical Report No.: SPANN-96.1*.

G David Forney. 1973. The viterbi algorithm. *Proceedings of the IEEE*, 61(3):268–278.

Tsu-Jui Fu, Peng-Hsuan Li, and Wei-Yun Ma. 2019. GraphRel: Modeling text as relational graphs for joint entity and relation extraction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1409–1418, Florence, Italy. Association for Computational Linguistics.

Nadia Ghamrawi and Andrew McCallum. 2005. Collective multi-label classification. In *Proceedings of the 14th ACM international conference on Information and knowledge management*, pages 195–200.

Zixia Jia, Zhaohui Yan, Haoyi Wu, and Kewei Tu. 2022. Span-based semantic role labeling with argument pruning and second-order inference. In *Proceedings of the AAAI Conference on Artificial Intelligence*. AAAI Press.

John Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016a. Neural architectures for named entity recognition. *arXiv preprint arXiv:1603.01360*.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016b. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270, San Diego, California. Association for Computational Linguistics.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*.

D. Khuê Lê-Huu and Karteek Alahari. 2021. Regularized frank-wolfe for dense crfs: Generalizing mean field and beyond. *arXiv preprint arXiv:2110.14759*.

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*.

Chao Lou, Songlin Yang, and Kewei Tu. 2022. Nested named entity recognition as latent lexicalized constituency parsing. In *ACL*.

Jie Lou, Yaojie Lu, Dai Dai, Wei Jia, Hongyu Lin, Xianpei Han, Le Sun, and Hua Wu. 2023. Universal information extraction as unified semantic matching. *arXiv preprint arXiv:2301.03282*.

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.

Yaojie Lu, Qing Liu, Dai Dai, Xinyan Xiao, Hongyu Lin, Xianpei Han, Le Sun, and Hua Wu. 2022. Unified structure generation for universal information extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5755–5772.

Yi Luan, Luheng He, Mari Ostendorf, and Hannaneh Hajishirzi. 2018. Multi-task identification of entities, relations, and coreference for scientific knowledge graph construction. *arXiv preprint arXiv:1808.09602*.

Yi Luan, Dave Wadden, Luheng He, Amy Shah, Mari Ostendorf, and Hannaneh Hajishirzi. 2019a. A general framework for information extraction using dynamic span graphs. *arXiv preprint arXiv:1904.03296*.

Yi Luan, Dave Wadden, Luheng He, Amy Shah, Mari Ostendorf, and Hannaneh Hajishirzi. 2019b. 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.

Yubo Ma, Zehao Wang, Yixin Cao, Mukai Li, Meiqi Chen, Kun Wang, and Jing Shao. 2022. Prompt for extraction? PAIE: Prompting argument interaction for event argument extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6759–6774, Dublin, Ireland. Association for Computational Linguistics.

T. Mikolov, I. Sutskever, C. Kai, G. Corrado, and J. Dean. 2013. Distributed representations of words and phrases and their compositionality.

Makoto Miwa and Yutaka Sasaki. 2014. Modeling joint entity and relation extraction with table representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1858–1869. ACL.

Trung Minh Nguyen and Thien Huu Nguyen. 2019. One for all: Neural joint modeling of entities and events. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 6851–6858. AAAI Press.

Giovanni Paolini, Ben Athiwaratkun, Jason Krone, Jie Ma, Alessandro Achille, Rishita Anubhai, Cícero Nogueira dos Santos, Bing Xiang, and Stefano Soatto. 2021. Structured prediction as translation between augmented natural languages. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Changzhi Sun, Yeyun Gong, Yuanbin Wu, Ming Gong, Daxin Jiang, Man Lan, Shiliang Sun, and Nan Duan. 2019. Joint type inference on entities and relations via graph convolutional networks. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1361–1370, Florence, Italy. Association for Computational Linguistics.

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. *arXiv preprint arXiv:2103.09330*.

Minh Van Nguyen, Bonan Min, Franck Dernoncourt, and Thien Nguyen. 2022a. 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.

Minh Van Nguyen, Bonan Min, Franck Dernoncourt, and Thien Nguyen. 2022b. Learning cross-task dependencies for joint extraction of entities, events, event arguments, and relations. In *EMNLP 2022*.

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, Medero Julie, and Kazuaki Maeda. 2005. ACE 2005 multilingual training corpus. In *Linguistic Data Consortium*.

Haoyu Wang, Ming Tan, Mo Yu, Shiyu Chang, Dakuo Wang, Kun Xu, Xiaoxiao Guo, and Saloni Potdar. 2019a. Extracting multiple-relations in one-pass with pre-trained transformers. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1371–1377, Florence, Italy. Association for Computational Linguistics.

Jue Wang and Wei Lu. 2020. Two are better than one: Joint entity and relation extraction with table-sequence encoders. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1706–1721, Online. Association for Computational Linguistics.

Wenya Wang and Sinno Jialin Pan. 2020. Integrating deep learning with logic fusion for information extraction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 9225–9232.

Wenya Wang and Sinno Jialin Pan. 2021. Variational deep logic network for joint inference of entities and relations. *Computational Linguistics*, pages 1–38.

Xinyu Wang, Jingxian Huang, and Kewei Tu. 2019b. Second-order semantic dependency parsing with end-to-end neural networks. *arXiv preprint arXiv:1906.07880*.

Xinyu Wang and Kewei Tu. 2020. Second-order neural dependency parsing with message passing and end-to-end training. *arXiv preprint arXiv:2010.05003*.

Yijun Wang, Changzhi Sun, Yuanbin Wu, Hao Zhou, Lei Li, and Junchi Yan. 2021. Unire: A unified label space for entity relation 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 220–231.

Eric P Xing, Michael I Jordan, and Stuart Russell. 2012. A generalized mean field algorithm for variational inference in exponential families. *arXiv preprint arXiv:1212.2512*.

Zhiheng Yan, Chong Zhang, Jinlan Fu, Qi Zhang, and Zhongyu Wei. 2021. A partition filter network for joint entity and relation extraction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 185–197. Association for Computational Linguistics.

Deming Ye, Yankai Lin, Peng Li, and Maosong Sun. 2022. Packed levitated marker for entity and relation extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4904–4917.

Juntao Yu, Bernd Bohnet, and Massimo Poesio. 2020. Named entity recognition as dependency parsing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6470–6476, Online. Association for Computational Linguistics.

Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, and Jun Zhao. 2014. Relation classification via convolutional deep neural network. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 2335–2344, Dublin, Ireland. Dublin City University and Association for Computational Linguistics.

Tongtao Zhang, Heng Ji, and Avirup Sil. 2019. Joint entity and event extraction with generative adversarial imitation learning. *Data Intell.*, 1(2):99–120.

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.

Shuai Zheng, Sadeep Jayasumana, Bernardino Romera-Paredes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, and Philip HS Torr. 2015. Conditional random fields as recurrent neural networks. In *Proceedings of the IEEE international conference on computer vision*, pages 1529–1537.

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, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 1227–1236. Association for Computational Linguistics.

Zexuan Zhong and Danqi Chen. 2020. A frustratingly easy approach for entity and relation extraction. *arXiv preprint arXiv:2010.12812*.

A Details on Identification Module

A multi-layer perceptron (MLP) takes word representations $H = [\mathbf{h}_1, ..., \mathbf{h}_n]$ as input and outputs an emission score $\mathbf{u}_i$ for each word. With a learnable transition score matrix A, a labeled sequence $\mathbf{y} = (y_1, ..., y_n)$ can be scored as $s(\mathbf{y}, H) = \sum_{i=1}^{n} (\mathbf{u}_i)_{y_i} + A_{y_{i-1}, y_i}$.

Inference We use the Viterbi algorithm (Forney, 1973) to obtain the sequence that has the highest score: $\hat{\mathbf{y}} = \arg\max_{\mathbf{y}} s(\mathbf{y}, H)$. Then we select the spans whose inter-words are labeled as B-X and I-X in the optimal output sequence as predicted node set.

Learning We maximize the probability of the target sequence to learn the identification module.

$$P(\mathbf{y}^*|\mathbf{w}) = \frac{\exp(s(\mathbf{y}^*, H))}{\sum_{\mathbf{y}'} \exp(s(\mathbf{y}', H))} = \frac{1}{\mathcal{Z}} \exp(s(\mathbf{y}^*, H))$$

where $\mathbf{y}^*$ is the target sequence and $\mathcal{Z}$ is the partition function. We can use the forward-backward algorithm (Dugad and Desai, 1996) to calculate $\mathcal{Z}$.

Of note, we did not consider nested spans in this work, which can easily be adopted to our framework using similar methods as in Yu et al. (2020); Lou et al. (2022) to identify graph nodes if span nesting.

B Hyper-parameters

For the hidden sizes of unary FNNs and most optimizer parameters, we use the default hyper-parameters following (Lin et al., 2020). The hidden sizes of FNNs in high-order scoring are tuned between $\{150, 300\}$. The iteration step T of MFVI is tuned between $\{1, 2, 3\}$, and it is set to 1 or 2 in different settings. We choose the hyper-parameters according to the performance of the development set after 80 epoch runs. The main hyper-parameters are listed in Table 8.

C Experimental results on ACE05-E given gold entities and triggers

Table 9 shows the experimental results on ACE05-E given gold entities and triggers. We can find

Setting	Value
Unary scoring	
FNN(entity)	150
FNN(trigger)	600
FNN(relation)	150
FNN(role)	600
Binary scoring	
FNN(head)	150
FNN(tail)	150
FNN(mid)	150
Ternary scoring	
FNN(head)	150
FNN(tail)	150
Other setting	
batch size	10
dropout rate	0.4
learning rate of Pretrained LM encoder	1e-5
lr decay of Pretrained LM encoder	1e-5
learning rate of other modules	1e-3
lr decay of other modules	1e-3
warm-up epochs	5
total epochs	80
gradient clipping	5.0

Table 8: Summary of hyper-parameters

	Ent	*Tri-C*	*Arg-I*	*Arg-C*
CRFIE baseline	96.0	93.1	70.7	68.3
CRFIE *homo* (+sib)	96.0	93.6	72.0	69.2
CRFIE *hete* (+ter)	95.9	94.1	71.7	69.2
CRFIE *homo+hete* (+sib+ter)	96.0	93.6	72.3	69.4

Table 9: Average F1 on ACE05-E dataset. The gold triggers and entities are given.

	Ent	*Tri-I*	*Tri-C*	*Arg-I*	*Arg-C*
CRFIE baseline	90.8	77.7	74.8	58.5	56.4
CRFIE *homo* (+sib)	90.6	77.7	74.5	59.1	57.1
CRFIE *homo* (+sib+cop)	90.8	77.7	74.6	58.7	57.1
CRFIE *hete* (+ter)	90.7	77.7	74.3	59.2	57.2
CRFIE *homo+hete* (+sib+ter)	90.6	77.7	74.3	59.6	57.5

Table 10: Average F1 on ACE05-E dataset with encoders of BERT-large-cased

that without the error of the identification module, the performance gap between our baseline and high-order models further increases, and using both sibling factors and ternary factors improves further.

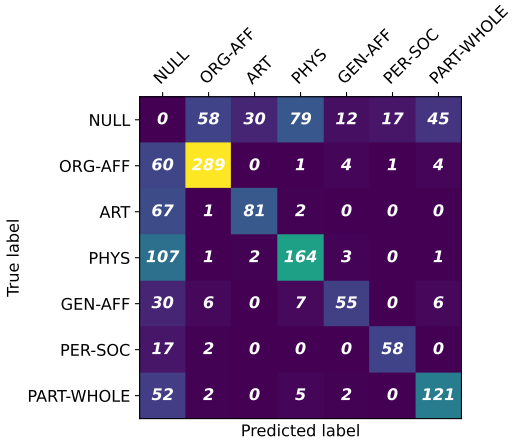
D Ablation Study

We show the experimental results of different factor combinations on Table 10, Table 11 and Table 12.

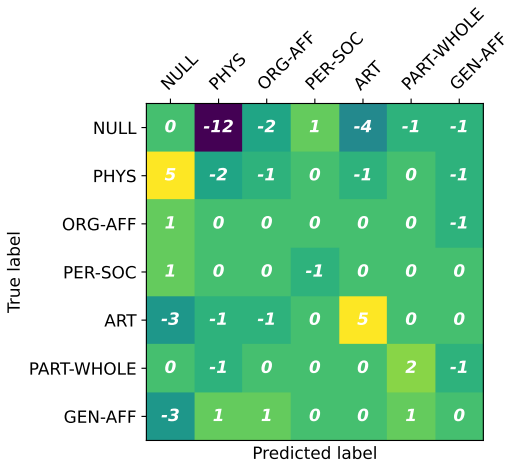
On Table 12, *role-sib* represents sib of role pairs, *rel-sib* represents sib of relation pairs, and *r+r-sib* represents sib of both role pairs and relation pairs. The *hete (+cop), hete (+gp), hete (+cop+gp)* are in ***hete* case (i)**.

E Error Correction Analysis

We take joint EntR and RelE as an example to show the number of error corrections of our high-order



(a) Our baseline



(b) Error correction matrix

Figure 5: Confusion matrix of the relation types. (a) The relation numbers of our baseline model on predicted entities. (b) The correction numbers of our high-order model relative to the baseline model. We do not have statistics on Null-Null.

	Ent	*Rel*	*Rel+*
CRFIE baseline	89.8	69.9	67.5
CRFIE *homo* (+sib)	90.0	70.8	68.1
CRFIE *homo* (+cop)	90.1	70.1	68.0
CRFIE *homo* (+gp)	90.2	70.0	67.7
CRFIE *homo* (+sib+cop)	90.2	70.8	68.2
CRFIE *hete* (+ter)	90.1	70.4	68.3

Table 11: Average F1 on ACE05-R dataset with encoder of ALBERT-XXLarg-v1

model compared to our baseline model in terms of relation types. From Fig. 5, we can find that our high-order model corrects the errors of our baseline model in relation types (the numbers are expected to be positive in the diagonal and to be negative otherwise).

	Ent	*Rel*	*Tri-I*	*Tri-C*	*Arg-I*	*Arg-C*
CRFIE baseline	90.8	65.3	77.4	74.6	60.0	58.1
CRFIE *homo* (role-sib)	90.8	65.1	77.4	74.6	60.3	58.4
CRFIE *homo* (rel-sib)	91.0	65.6	77.4	74.8	60.1	58.5
CRFIE *homo* (r+r-sib)	90.9	65.4	77.4	74.8	60.1	58.3
CRFIE *hete* (+cop)	90.7	65.9	77.4	74.6	60.3	58.2
CRFIE *hete* (+gp)	90.7	65.8	77.4	75.1	60.8	59.0
CRFIE *hete* (+cop+gp)	90.7	65.1	77.4	74.8	60.3	58.5
CRFIE *homo* case (i) + *homo* case (ii)	90.9	65.4	77.4	74.8	60.1	58.3

Table 12: Average F1 on ACE05-E+ dataset. All models use BERT-large-cased encoder.

F Re-evaluation of PL-Marker

For the relation extraction task, some corpus have symmetric relations, meaning the ordering of the two entities does not matter (e.g., 'PER-SOC' in ACE2005). A symmetric relation is only annotated in one direction in the annotation data. PL-Marker counts a symmetric relation twice both for prediction number and gold number, but other work only counts once for the prediction and gold numbers.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
Limitation

☒ A2. Did you discuss any potential risks of your work?
We do regular NLP task and use standard NLP datasets

☑ A3. Do the abstract and introduction summarize the paper's main claims?
abstract, 1 introduction

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☒ Did you use or create scientific artifacts?

Left blank.

☐ B1. Did you cite the creators of artifacts you used?
Not applicable. Left blank.

☐ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
Not applicable. Left blank.

☐ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
Not applicable. Left blank.

☐ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
Not applicable. Left blank.

☐ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Not applicable. Left blank.

☐ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
Not applicable. Left blank.

C ☑ Did you run computational experiments?

3 Experiments

☒ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
We use pretrained language model as encoder as other work and the number of parameters of other part in our model is in a much smaller scale.

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Appendix C

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
3 Experiments

☒ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
Left blank.

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
Not applicable. Left blank.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
Not applicable. Left blank.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
Not applicable. Left blank.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
Not applicable. Left blank.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
Not applicable. Left blank.