HighIE: High-Order Inference for Entity Recognition, Relation Extraction, and Event Extraction

Anonymous ACL submission

Abstract

Most prior work on information extraction (IE) typically predicts labels of individual instances (*e.g.*, event triggers, relations, entities) independently regardless of their interactions. We propose a novel framework, HighIE, that aims to integrate high-order cross-subtask and cross-instance dependencies in both learning and inference. High-order inference on label variables is an NP-hard problem. To address it, we propose a high-order decoder that is unfolded from an approximate inference algorithm. The experimental results show that our approach achieves consistent improvement compared with prior work.[1]

1 Introduction

Information Extraction (IE) is the task of extracting structured information from unstructured texts. It is comprised of various subtasks, such as entity recognition, coreference resolution, relation extraction, and event extraction. Conventional approach construct IE models by solely based on local features. Recent advances point out that high-order interactions (*e.g.*, cross-subtask and cross-instance interactions) among different instances (*e.g.*, event triggers, relations, entities) can provide rich information in various IE subtasks (Li et al., 2014; Miwa and Sasaki, 2014; Yang and Mitchell, 2016; Kirschnick et al., 2016; Luan et al., 2018, 2019; Wadden et al., 2019; Lin et al., 2020). Specifically, types of entities can provide information that is useful to predict their relations or limit the roles they play in some events. Similarly, a relation between two entities would restrict the types of the entities. Take event extraction as an example, in event `DIE`, a `PERSON` entity is more likely to play a role of `VICTIM`; two entities having `LIVEIN` relation are more likely to be `PERSON` and `LOCATION` types.

Some recent work has demonstrated improvements on local features by leveraging interactions between different instances or different subtasks. For example, Luan et al. (2018) implicitly leverages information from other tasks by jointly training multiple sub-tasks with shared embeddings; Luan et al. (2019) and Wadden et al. (2019) use a dynamic span graph to update span representations according to the prediction of task-specified relations. However, these methods only implicitly model interactions via representation learning. Different from them, Lin et al. (2020) designs a template to generate global features and takes them as global constraints in the training process. However, their global features require human design and cannot be directly used in the inference process. Wang and Pan (2021) try to learn intensive correlations between labels variables in inference by designing a logic network, but they cannot achieve end-to-end training, which leads to unsatisfactory performance.

In this work, we aim to automatically integrate high-order interactions across instances or subtasks and propose a novel model named HighIE that can automatically incorporate high-order correlations across sub-tasks and sub-instances in both training and inference. HighIE consists of two modules: the identification module and the classification module. We first train the identification module to identify triggers and entities in a sentence [2], then fix the identification module and take the predictions as input to train a classification module with high-order interactions to jointly predict the labels of all instances. Exact high-order inference in this situation is NP-hard, so we design a neural decoder that is unfolded from mean-field variational inference (Zheng et al., 2015; Wang et al.,

[1] The code and trained models can be found at `https://anonymous`.

[2] Different from most previous work that jointly trains the identification and classification modules, we empirically find that separate training achieves stabler and better performance. The same observation has been demonstrated by Zhong and Chen (2020) in relation extraction.

2019; Wang and Tu, 2020) to achieve approximate inference and end-to-end training. Experimental results show that our approach achieves significant improvement over previous work.

2 Approach

Task Formulation Given a sentence $\mathbf{w} = \{w_i : i = 1, .., n\}$ with n words,

▷ Entity recognition (ER) aims to identify some spans in a sentence as entities and label entity types, $e_i = (a_i, b_i, l^{\text{E}})$ denotes the i-th entity with entity type l^{E} consisting of $w_{a_i}, .., w_{b_i}$ words.

▷ Relation extraction (RE) aims to label relations between entity pairs, $r_{ij}^{\text{RE}} = (i, j, l^{\text{RE}})$ denotes the directed relation from the entity e_i to the entity e_j, with l^{RE} being the relation type.

▷ Event extraction (EE) aims to label event types, triggers and argument roles. We take the event type l^{T} as the label of its trigger and thus the prediction targets are triggers $\{t_j = (a_j, b_j, l^{\text{T}})\}$, denoting the j-th trigger consisting of $w_{a_j}, .., w_{b_j}$ words with l^{T} as its label, and roles $\{r_{ji}^{\text{TE}} = (j, i, l^{\text{TE}})\}$, each denoting a relation between the j-th trigger and the i-th entity (argument) with l^{TE} as the role type that the argument plays.

Encoder Both the identification network and classification network adopt pre-trained transformer-based encoders. For the i-th word, we use the average of all sub-word embeddings extracted from the encoder as the word's representation $\mathbf{h}_i^t$, where $t \in \{I, C\}$ and I represents the identification task while C represents the classification task.

2.1 Identification Module

Following previous work (Lin et al., 2020), we formulate event trigger identification and entity identification as sequence labeling tasks with a BIO schema. We use a linear-chain conditional random field (CRF) (Lafferty et al., 2001) as a decoder to predict the optimal output sequence. Details refer Appendix A.

2.2 Classification Module

The classification module predicts event types (labels of triggers), entity types, existence and types of roles between trigger-entity pairs, and existence and labels of relations between entity-entity pairs. We consider high-order interactions and the joint inference between ER and EE tasks, and between ER and RE tasks.

We first obtain the trigger representation $\mathbf{z}_j^{\text{T}} = \text{Avg}(\mathbf{h}_{a_j}^C, ..., \mathbf{h}_{b_j}^C)$ and entity representation $\mathbf{z}_i^{\text{E}} = \text{Avg}(\mathbf{h}_{a_i}^C, ..., \mathbf{h}_{b_i}^C)$ by averaging all word representations in the j-th trigger and i-th entity respectively.

We directly input the trigger representation into an MLP to obtain the first-order scores of possible event types $\mathbf{s}_j^{\text{T}} = \text{MLP}_1(\mathbf{z}_j^{\text{T}})$, where $\mathbf{s}_j^{\text{T}}$ is a R_1-dimensional vector with R_1 being the number of possible event types. In a similar way, the first-order scores of possible entity types are $\mathbf{s}_i^{\text{E}} = \text{MLP}_2(\mathbf{z}_i^{\text{E}})$, where $\mathbf{s}_i^{\text{E}}$ is a R_2-dimensional vector with R_2 being the number of possible entity types. For role prediction between trigger-entity pairs, the trigger representation and entity representation are concatenated and then fed into an MLP to obtain the first-order scores $\mathbf{s}_{ji}^{\text{TE}} = \text{MLP}_3([\mathbf{z}_j^{\text{T}}; \mathbf{z}_i^{\text{E}}])$, where $\mathbf{s}_{ji}^{\text{TE}}$ is R_3-dimensional vector with R_3 being the number of possible role types, including an additional "None" label denoting that the entity is not an argument of the event. Similarly, for relation prediction between entity-entity pairs, we calculate the first-order relation scores $\mathbf{s}_{ij}^{\text{RE}} = \text{MLP}_4([\mathbf{z}_i^{\text{E}}; \mathbf{z}_j^{\text{E}}])$, where $\mathbf{s}_{ij}^{\text{RE}}$ is R_4-dimensional vector with R_4 being the number of possible relation types, including an additional "None" label.

To leverage high-order interactions between different instances of different sub-tasks, we design two classes of high-order factors: trigger-role-entity (tre) factors to model the correlations between event types, entity types and role types; entity-relation (er) factors to model the correlations of entity types and relation types. We use a decomposed penta-linear function f_{pen} and tri-linear function f_{tri} to calculate the high-order scores as follows:

$$\boldsymbol{S}_{j,ji,i}^{(\text{tre})} = f_{pen}(\mathbf{z}_j^{\text{T}}, \mathbf{z}_i^{\text{E}}, \mathbf{g}_j^{\text{T}}, \mathbf{g}_{ji}^{\text{TE}}, \mathbf{g}_i^{\text{E}}) \tag{158}$$

$$= \sum_{m=1}^{k} \mathbf{z}'_j \circ \mathbf{z}'_i \circ \mathbf{g}_j^{\text{T}} \circ \mathbf{g}_{ji}^{\text{TE}} \circ \mathbf{g}_i^{\text{E}} \tag{159}$$

$$S_{i,ij}^{(\text{er})} = f_{tri}(\mathbf{z}_i^{\text{E}}, \mathbf{g}_i^{\text{E}}, \mathbf{g}_{ij}^{\text{RE}}) = \sum_{m=1}^{k} \mathbf{z}''_i \circ \mathbf{g}_i^{\text{E}} \circ \mathbf{g}_{ij}^{\text{RE}} \tag{160}$$

$$\mathbf{z}'_j = U_1 \mathbf{z}_j^{\text{T}} \quad \mathbf{z}'_i = U_2 \mathbf{z}_i^{\text{E}} \quad \mathbf{z}''_i = U_3 \mathbf{z}_i^{\text{E}} \tag{161}$$

where $U_1, ..., U_3$ are three $(d \times k)$-dimensional matrices. d is the hidden size of the input representations. k is the decomposition rank. $\mathbf{g}_j^{\text{T}}$, $\mathbf{g}_i^{\text{E}}$, $\mathbf{g}_{ji}^{\text{TE}}$, and $\mathbf{g}_{ij}^{\text{RE}}$ are learnable parameter matrix with sizes $(R_1 \times k)$, $(R_2 \times k)$, $(R_3 \times k)$, and $(R_4 \times k)$ respectively. $\boldsymbol{S}_{j,ji,i}^{(\text{tre})}$ is $(R_1 \times R_3 \times R_2)$-dimensional

tensor and $S^{(\text{er})}_{i,ij}$ is $(R_2 \times R_4)$-dimensional matrix. $\circ$ denotes element-wise product.

Inference For first-order inference without considering the high-order scores, we adopt the same beam decoding as in Lin et al. (2020). For high-order inference, we predict all the entity types, trigger types, roles and relations by maximizing the sum of their first-order scores and high-order scores. Since it is an NP-hard problem, we use Mean Field Variational Inference (MFVI) (Xing et al., 2012) for approximate inference. MFVI updates approximate posterior marginal distributions $Q(\cdot)$ according to messages $\mathcal{F}(\cdot)$ passed by variables in the same factors. For joint ER and EE tasks, the messages are calculated as follows:

$$\begin{aligned}
\mathcal{F}_j^{\text{T}(t-1)} &= \sum_i \sum_{l^{\text{E}}} Q_i^{\text{E}(t-1)}(l^{\text{E}}) \\
&\quad \Big(\sum_{l^{\text{TE}}} Q_{ji}^{\text{TE}(t-1)}(l^{\text{TE}})(\boldsymbol{S}^{(\text{tre})}_{j,ji,i})_{l^{\text{TE}}}\Big)_{l^{\text{E}}} \\
\mathcal{F}_i^{\text{E}(t-1)} &= \sum_j \sum_{l^{\text{T}}} Q_j^{\text{T}(t-1)}(l^{\text{T}}) \\
&\quad \Big(\sum_{l^{\text{TE}}} Q_{ji}^{\text{TE}(t-1)}(l^{\text{TE}})(\boldsymbol{S}^{(\text{tre})}_{j,ji,i})_{l^{\text{TE}}}\Big)_{l^{\text{T}}} \\
\mathcal{F}_{ji}^{\text{TE}(t-1)} &= \sum_{l^{\text{T}}} Q_j^{\text{T}(t-1)}(l^{\text{T}}) \\
&\quad \Big(\sum_{l^{\text{E}}} Q_i^{\text{E}(t-1)}(l^{\text{E}})(\boldsymbol{S}^{(\text{tre})}_{j,ji,i})_{l^{\text{E}}}\Big)_{l^{\text{T}}}
\end{aligned}$$

Then the posteriors $Q(\cdot)$ are updated according to the messages $\mathcal{F}(\cdot)$.

$$\begin{aligned}
Q_j^{\text{T}(t)} &\propto \exp\{\mathbf{s}_j^{\text{T}} + \mathcal{F}_j^{\text{T}(t-1)}\} \\
Q_i^{\text{E}(t)} &\propto \exp\{\mathbf{s}_i^{\text{E}} + \mathcal{F}_i^{\text{E}(t-1)}\} \\
Q_{ji}^{\text{TE}(t)} &\propto \exp\{\mathbf{s}_{ji}^{\text{TE}} + \mathcal{F}_{ji}^{\text{TE}(t-1)}\}
\end{aligned}$$

The joint ER and RE tasks deploy a similar process:

$$\begin{aligned}
\mathcal{F}_{ij}^{\text{RE}(t-1)} &= \sum_{l^{\text{E}}} Q_i^{\text{E}(t-1)}(l^{\text{E}})(S^{(\text{er})}_{i,ij})_{l^{\text{E}}} \\
&\quad + \sum_{l^{\text{E}}} Q_j^{\text{E}(t-1)}(l^{\text{E}})(S^{(\text{er})}_{j,ij})_{l^{\text{E}}} \\
\mathcal{F}_i^{\text{E}(t-1)} &= \sum_{j \neq i} \sum_{l^{\text{RE}}} Q_{ij}^{\text{RE}(t-1)}(l^{\text{RE}})(S^{(\text{er})}_{i,ij})_{l^{\text{RE}}} \\
Q_i^{\text{E}(t)} &\propto \exp\{\mathbf{s}_i^{\text{E}} + \mathcal{F}_i^{\text{E}(t-1)}\} \\
Q_{ij}^{\text{RE}(t)} &\propto \exp\{\mathbf{s}_{ij}^{\text{RE}} + \mathcal{F}_{ij}^{\text{RE}(t-1)}\}
\end{aligned}$$

The initial distribution $Q^{(0)}$ is set by normalizing exponentiated first-order scores. After a fixed N number of iterations, we obtain the posteriors $Q^{(N)}$. Then we use beam search to get the predictions of all items according to the posteriors.

Learning The training target is to maximize the probability of the ground truth. We do multi-task learning with cross entropy losses as follows:

$$\begin{aligned}
\mathcal{L} &= -\sum_i \log P(\hat{l}_i^{\text{E}}|\mathbf{w}) - \sum_j \log P(\hat{l}_j^{\text{T}}|\mathbf{w}) \\
&\quad - \sum_{ji} \log P(\hat{l}_{ji}^{\text{TE}}|\mathbf{w}) - \sum_{ij} \log P(\hat{l}_{ij}^{\text{RE}}|\mathbf{w}) \\
P(\hat{l}^{\text{X}}|\mathbf{w}) &= Q^{\text{X}(N)}(\hat{l}^{\text{X}})
\end{aligned}$$

where $\text{X} \in \{\text{T}, \text{E}, \text{TE}, \text{RE}\}$ and $\hat{l}$ is the target labels of each task.

3 Experiments

Dataset We evaluate our model on the ACE2005 corpus (Walker et al., 2005) which provides the entity, relation, and event annotations for different languages including English, Chinese, and Arabic. Following (Lu et al., 2021; Lin et al., 2020; Wadden et al., 2019), we conduct experiments on two English datasets: ACE05-R for ER and RE and ACE05-E and ACE05-E+ for ER and EE, 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 these datasets. Detailed data statistics are provided in Appendix B.

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). For the ER task, an entity (*Ent*) is correct if both its type and offsets match a gold entity. For the RE task, a relation (*Relation*) is correct if both its type and the offsets of its related entities match a gold relation. 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.

Implementation Details For fair comparison with the state-of-the-art system (Lin et al., 2020), we use the bert-large-cased model (Devlin et al., 2018) as our encoder for ACE2005 English datasets. We train our model with BERTAdam optimizer. All hyper-parameters of our base model are the same as in Lin et al. (2020). For our high-order model, we set the decomposed size $k = 150$ and the iteration number $N = 3$. Detailed hyper-parameter values are provided in Appendix C.

	Ent	*Trig-I*	*Trig-C*	*Arg-I*	*Arg-C*
DYGIE++	89.7	-	69.7	53.0	48.8
ONEIE	90.2	**78.2**	**74.7**	59.2	56.8
T2E	-	-	71.9	-	53.8
HighIE	**91.1**	77.9	**74.7**	**59.6**	**57.6**

Table 1: F1 scores averaged on three runs on the ACE05-E dataset. We use tre factor in this setting.

	Ent	*Trig-I*	*Trig-C*	*Arg-I*	*Arg-C*
ONEIE	89.6	75.6	72.8	57.3	54.8
T2E	-	-	71.8	-	54.5
HighIE	**90.7**	**76.5**	**73.9**	**60.0**	**58.1**

Table 2: F1 scores on the ACE05-E+ dataset. We use tre factor in this setting.

	DYGIE++	ONEIE	HighIE
Entity	88.6	88.8	**89.0**
Relation	63.4	67.5	**68.5**

Table 3: F1 scores on the ACE05-R dataset. We use er factor in this setting.

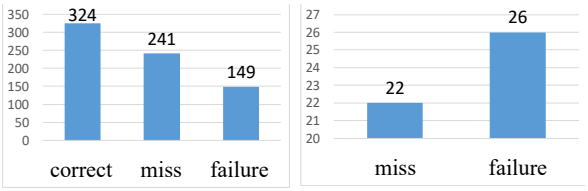
Comparisons Some previous work of event extraction leveraged gold triggers or gold entities. However, our approach assumes that all the triggers and entities are unreachable and should be predicted. Besides, we jointly train the different tasks (ER, RE, and EE) in the ACE05-E and ACE05-E+ datasets. Therefore, we compare our approach with the following models in the same settings: (1) DYGIE++ (Wadden et al., 2019) which is a general multi-task IE framework learning dynamic span representation, (2) OneIE (Lin et al., 2020) which extends DYGIE++ by incorporating global features, (3) TEXT2EVENT (denoted by T2E) (Lu et al., 2021) which formulates event extraction as a sequence-to-structure generation paradigm to share knowledge between different components.

3.1 Results and Analyses

Main Results Table 1, Table 2 and Table 3 show the experimental results of our approach on ACE05-E, ACE05-E+ and ACE05-R, respectively. We only use tre factor in the experiments on ACE05-E and ACE05-E+ datasets, thus we only report the performance of ER and EE tasks. In this situation, the RE is trained as auxiliary task and its performance is comparable with previous work. ACE05-R only contains entity and relation annotations and we only train ER and RE tasks on this dataset. We can find that our HighIE performs better in most cases. Despite the F1 score of *Trig-I* (predicted by the identification module) in Table 1 is a bit lower than ONEIE, the F1 scores of *Arg-C* is still higher than ONEIE. Note that the *Arg-C* is the most difficult sub-task in EE because it requires that the argument offsets, the roles and the corresponding event types are all correctly predicted. Therefore, our good performance on *Arg-C* proves that the intensive high-order interactions between variables are beneficial to label classifications.

	Ent	*Trig-I*	*Trig-C*	*Arg-I*	*Arg-C*
Joint	90.7	77.0	74.4	58	55.0
Our base	91.1	77.9	74.3	58.3	56.2
HighIE	91.1	77.9	74.7	59.6	57.6

Table 4: Comparison of separate and joint learning of identification module on the ACE05-E dataset.



(a) Baseline predictions statistics of arguments

(b) Argument classification error fixed by high-order model

Figure 1: Error type 'failure' means the model assigns a wrong role of an identified argument and 'miss' means the model assigns a gold argument with a "None" label.

Analysis of Separate Learning In Tabel 4, we compare the joint learning baseline (Joint), the separate learning baseline (Our base), and our high-order model (HighIE) on the ACE05-E dataset. It can be seen that training the identification module independently from the classification module performs better on most sub-tasks than jointly training. With high-order inference, the results get further improvement.

Error Fixed by High-order Model We count the number of different errors in argument classification (*Arg-C*) of our base model and the number of error corrections by our HighIE model in Fig.1. For 'failure' error, we only consider the arguments with correct span boundaries to exclude the error from upstream identification module. A case study is provided in Appendix E.

4 Conclusion

In this paper, we propose a novel framework that can leverage high-order interactions between different instances and different IE sub-tasks. Our framework consists of identification module to identify event triggers and entities and classification module with high-order inference to label all instances. Experimental results show that our high-order approach achieves consistent improvement over previous work.

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 Artifical 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.

Johannes Kirschnick, Holmer Hemsen, and Volker Markl. 2016. Jedi: Joint entity and relation detection using type inference. In *Proceedings of ACL-2016 System Demonstrations*, pages 61–66.

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

Qi Li, Heng Ji, Yu Hong, and Sujian Li. 2014. Constructing information networks using one single model. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1846–1851.

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.

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, 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. 2019. A general framework for information extraction using dynamic span graphs. *arXiv preprint arXiv:1904.03296*.

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)*, pages 1858–1869.

David Wadden, Ulme Wennberg, Yi Luan, and Hannaneh Hajishirzi. 2019. Entity, relation, and event extraction with contextualized span representations. *arXiv preprint arXiv:1909.03546*.

Christopher Walker, Stephanie Strassel, Medero Julie, and Kazuaki Maeda. 2005. Ace 2005 multilingual training corpus.

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. 2019. 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*.

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*.

Bishan Yang and Tom Mitchell. 2016. Joint extraction of events and entities within a document context. *arXiv preprint arXiv:1609.03632*.

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.

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

A Identification Module

A multi-layer perceptron (MLP) takes word representations $H^I = [\mathbf{h}_1^I, ..., \mathbf{h}_n^I]$ 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 $s(\mathbf{y}, H^I) = \sum_{i=1}^{n} (\mathbf{u}_i)_{y_i} + A_{y_{i-1}, y_i}$.

Inference We use the Viterbi algorithm (Forney, 1973) to obtain the sequence that has the highest score: $\hat{\mathbf{y}} = \arg\max_{\mathbf{y}} s(\mathbf{y}, H^I)$.

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

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

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 $\mathcal{Z}$.

B Dataset Statistics

Statistics of all datasets we used are shown in Tabel 5.

	Split	#Sentences	#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	19,240	47,525	7,152	4,419
	Dev	902	3,422	728	468
	Test	676	3,673	802	424

Table 5: Datasets statistics

C Hyper-parameters

We use the default hyper-parameters following (Lin et al., 2020). The main hyper-parameters are listed in Table 6.

D Comparison of Base Model and HighIE

Tabel 7 and Tabel 8 give the results comparison of our base model and HighIE on ACE05-E+ and ACE05-R datasets.

E Case Study

Fig. 2 offers an example where our high-order model fixes a wrong prediction by the baseline model. In the sentence, 'bombed' is the trigger and 'bridges' is the corresponding candidate argument. The baseline model predicts 'bridges' to take the 'Target' role while the gold role should be 'Place'.

Network	**Hidden size**
FFN(ER)	2*150
FFN(RE)	2*150
FFN(EE)	2*600
Decomposed rank	150
Iteration of inference	3
Dropouts	**Dropout rate**
FFN	0.4
Optimizer	
Learning rate of BERT	5e-5
LR decay of BERT	1e-5
Learning rate of other parameters	1e-3
LR decay of other parameters	1e-3
Grad clipping	5.0

Table 6: Summary of hyper-parameters

	Ent	*Trig-I*	*Trig-C*	*Arg-I*	*Arg-C*
Our base	90.6	76.5	74.1	59.2	57.6
HighIE	90.7	76.5	73.9	60.0	58.1

Table 7: Comparison of our base model and HighIE on the ACE05-E+ dataset.

	ACE05-R	
	Ent	*Rel*
Our base	88.8	67.7
HighIE	89.0	68.5

Table 8: Comparison of our base model and HighIE on the ACE05-R dataset.

U.S. aircraft ***bombed*** Iraqi tanks holding bridges close to the city.

Gold
Entities: 1) 'U.S.': GPE 2) 'aircraft': VEH 3) 'Iraqi': GPE 4) 'tanks': VEH 5) 'bridges': FAC 6) 'city': GPE
Events: 1) Attack: 'bombed->U.S. [Attacker]', 'bombed->aircraft [Instrument]', 'bombed->tanks [Target]', 'bombed->bridges [Place]'

Base
Events: 1) Attack: 'bombed->aircraft [Instrument]', 'bombed->tanks [Target]', 'bombed->bridges [Target]'

HighIE
Events: 1) Attack: 'bombed->aircraft [Instrument]', 'bombed->tanks [Target]', 'bombed->bridges [Place]'

Figure 2: Case study where our baseline model predicts a wrong role and our high-order model fixes the error.