Interactive Information Extraction by Semantic Information Graph

Siqi Fan[*1], **Yequan Wang**[*2], **Jing Li**[3], **Zheng Zhang**[4], **Shuo Shang**[†1] and **Peng Han**[†5]
[1]University of Electronic Science and Technology of China, Chengdu, China
[2]Beijing Academy of Artificial Intelligence, Beijing, China
[3]Intelligence, Abu Dhabi, United Arab Emirates
[4]Department of Computer Science and Technology, Tsinghua University, Beijing, China
[5]Aalborg University
{sqfann, tshwangyequan, zhangz.goal, jedi.shang}@gmail.com,
jingli.phd@hotmail.com, pengh@cs.aau.dk

Abstract

Information extraction (IE) mainly focuses on three highly correlated subtasks, *i.e.,* entity extraction, relation extraction and event extraction. Recently, there are studies using Abstract Meaning Representation (AMR) to utilize the intrinsic correlations among these three subtasks. AMR based models are capable of building the relationship of arguments. However, they are hard to deal with relations. In addition, the noises of AMR (*i.e.,* tags unrelated to IE tasks, nodes with unconcerned conception, and edge types with complicated hierarchical structures) disturb the decoding processing of IE. As a result, the decoding processing limited by the AMR cannot be worked effectively. To overcome the shortages, we propose an Interactive Information Extraction (InterIE) model based on a novel Semantic Information Graph (SIG). SIG can guide our InterIE model to tackle the three subtasks jointly. Furthermore, the well-designed SIG without noise is capable of enriching entity and event trigger representation, and capturing the edge connection between the information types. Experimental results show that our InterIE achieves state-of-the-art performance on all IE subtasks on the benchmark dataset (*i.e.,* ACE05-E+ and ACE05-E). More importantly, the proposed model is not sensitive to the decoding order, which goes beyond the limitations of AMR based methods.

1 Introduction

Information extraction (IE) aims to extract structured information from unstructured text. According to the definition from the ACE2005 program [Walker and Consortium, 2005], there are three IE subtasks, *i.e.,* entity extraction, relation extraction, and event extraction. In general, the three subtasks are regarded as an information network $G^I(V^I, E^I)$ construction [Li *et al.*, 2014]. Figure 1 shows the reentrancy and isolation features of the information network. Reentrancy means that a node may play a multi-role in an information network. As the name suggests, isolation denotes that entities or events exist in isolation. The two characteristics indicate that these three subtasks of IE are independent of each other and have potential connections.

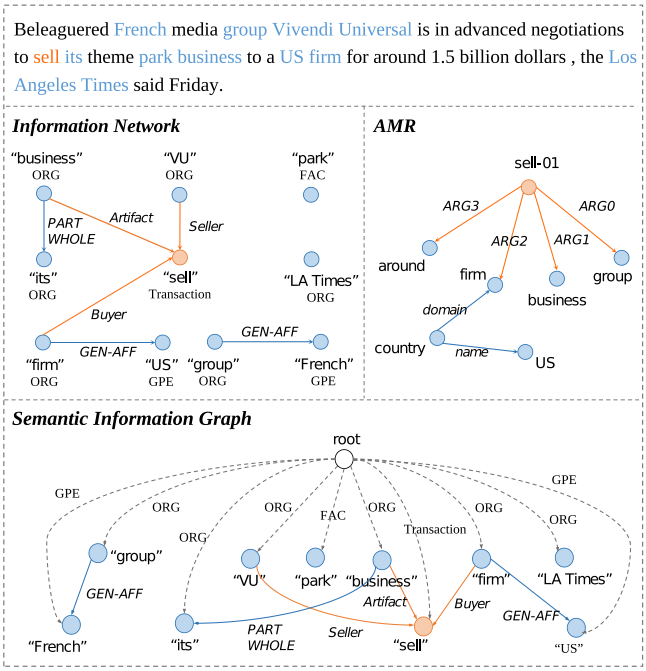


Figure 1: A comparison of information network, AMR graph and SIG for the same sentence from ACE05. Note that we only illustrate the part of AMR graph for this sentence due to the limited space, and "VU; LA Times" stands for "Vivendi Universal; Los Angeles Times ".

Existing studies rarely take into consideration the reentrancy feature. Early works solve the three subtasks separately [Li *et al.*, 2014], which leads to the error propagation problem and disallows interactions among subtasks during decoding. Then, deep learning-based methods [Wadden *et al.*, 2019; Luan *et al.*, 2019] are used to extract information separately. Further, an end-to-end IE framework is proposed

[*]Indicates equal contribution
[†]Corresponding Author

to incorporate global features during decoding to capture interactions between IE tasks [Lin *et al.*, 2020]. Recently, there are studies using Graph Neural Network (GNN) to do the subtasks and achieving promising performance. The dependency graph consumed by GNN is proposed to build an information network by restricting subtasks jointly [Nguyen *et al.*, 2021]. Despite the success of the above approaches, they are hard to utilize the correlation information of IE subtasks. We know that relation extraction is based on subject and object entities. Therefore, the recognition of subject and object entities is essential for relation extraction.

For argument extraction, an argument can be an entity, time expression, or value (*e.g.,* MONEY) [Lin *et al.*, 2020], and their extraction is the difficult point. Recently, Abstract Meaning of Representation (AMR) [Banarescu *et al.*, 2013] is used to guide the encoding and decoding process [Zhang and Ji, 2021]. AMRIE is proposed to manually assign the relation clusters of AMR to the label of IE tasks before putting them into graph attention networks [Zhang and Ji, 2021]. However, AMR is not designed for information extraction, so it cannot be used directly and effectively. Furthermore, although the AMR graph helps enrich the representation vectors of information network nodes, it also introduces redundant semantic information interference during the encoding and decoding stage.

To overcome those issues, we propose an Interactive Information Extraction (InterIE) model based on a novel Semantic Information Graph (SIG). SIG is able to guide the model to tackle the three subtasks jointly. Meanwhile, SIG is capable of reducing the noise information of AMR. Compared with AMR, the edge of SIG is the label of IE tasks, including relation and argument without transformation. Given an input text, we use the proposed InterIE to detect nodes using local classifiers [Lin *et al.*, 2020] firstly. Then, we map the candidate nodes to SIG and feed them to Graph Attention Networks (GATs) [Velickovic *et al.*, 2018], which enhances node representation for IE. Lastly, we use the SIG structure to determine the beam order and search for the optimal graph with global features during decoding. Experiments conducted on ACE05 dataset show that our InterIE model reaches the new state-of-the-art on all IE subtasks.

The major contributions of this paper are summarized as follows:

- We propose an Interactive Information Extraction (InterIE) with a novel Semantic Information Graph (SIG) to guide the IE subtasks jointly.
- Compared with AMR, SIG does not contain noise. Benefiting from this, InterIE could enrich entity and event trigger representations, and capture the edge connection between the information types through neighbors.
- We conduct experiments on benchmark datasets compared with strong baselines. Experiment results show that InterIE is competitive. More importantly, it is not sensitive to decoding order.

2 Approach

Figure 2 illustrates the overview of InterIE. Given an input text (*i.e.,* a sentence S), InterIE extracts the information network in three stages: i) SIG generation, generating the SIG for S based on sequence to graph transformer; ii) Graph aggregation, aggregating the candidate nodes and SIG through Graph Attention Networks (GATs); iii) Decoding processing, determining the decoding order and search the optimal graph with the global feature.

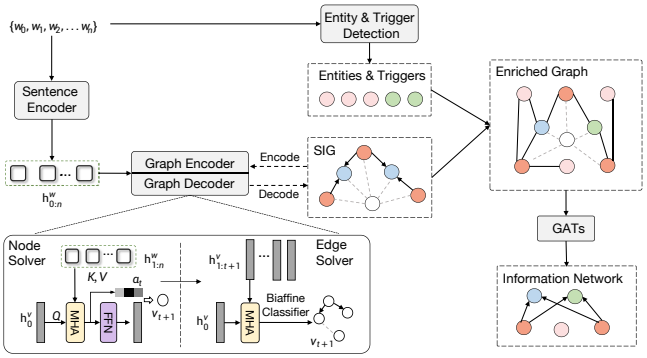


Figure 2: Overview of the InterIE framework. Given an input sentence, we insert a [START] token at the beginning and employ a sequence to graph transformer to generate the SIG iteratively. This generation process begins with a dummy node and terminates when a node with *end* label is obtained. We further enhance the generated SIG with entity and trigger nodes detected by CRFs. We update this enriched graph via GATs and finally parse it into an Information Network.

2.1 Problem Formulation

We formulate the subtasks of the joint information extraction problem as follows.

Entity Extraction is used to identify the span of the entity and assign pre-defined entity types to them. For example shown in Figure 1, "French" is recognized as a *GPE* entity.

Relation Extraction is used to assign a relation type to an ordered pair of entities. For example, the relation between "US" and "firm" is assigned as *GEN-AFF*.

Event Extraction is used to recognize the event and corresponding arguments. In general, it is divided into event identification and argument extraction. Event identification is the detection of event triggers (the words or phrases expressing the occurrences of the event clearly). As the name says, argument extraction denotes the extraction of arguments involved in the event (the roles participated in the event). For example, the word "sell" triggers a *Transaction* event, and the word "firm" denotes the *Buyer* argument while the phrase "Vivendi Universal" denotes the *Seller* argument.

These three IE subtasks can be formulated as an **information network**. Given an input sentence, our goal is to construct an information network $G^{info}(V^{info}, E^{info})$, where each node in V^{info} represents an entity or event trigger. Each edge in E^{info} represents a relation or argument role in events.

2.2 SIG Generation

As illustrated in Figure 1, the SIG $G(V, E)$ is a rooted, directed and labeled graph. The node of SIG represents an entity or an event trigger. The edge of SIG denotes label (*i.e.,*

entity type, relation type and argument). Different from the information network $G^{info}(V^{info}, E^{info})$, SIG equips with the virtual rooted structure which brings discrete IE subtasks together. We adopt the sequence to graph transformer to generate SIG graph incrementally. Next, we detail this generation process.

Graph Encoder. Given an input sentence $S = (w_1, \ldots, w_n)$ with a special [START] token w_0 at the beginning, we feed them into a transformer encoder to generate the corresponding embeddings $(h_0^w, h_1^w, \ldots, h_n^w)$. Here, h_0^w is the feature vector representing the whole sentence. Since the graph is sequentially generated, there is one new generated node v_t at step t. To trigger the generation process, we set a dummy node, marked as v_0. We feed these nodes $(v_0, v_1, \ldots, v_{t-1})$ into the transformer encoder with a masked attention mechanism to generate the updated graph node states $(h_{0,t}^v, h_{1,t}^v, \ldots, h_{t-1,t}^v)$. Here, $h_{0,t}^v$ is the representation of the graph at step t, and it is initialized as h_0^w. We put it into decoder to predict the source of graph node v_t and its relationships with previous nodes.

Node Solver. Given the graph state $h_{0,t}^v$ at step t and input sentence tokens $S = \{w_1, ..., w_n\}$, the node solver generates the node of SIG by deciding its position in the sentence or a pre-defined vocabulary (the set of all entities and triggers in training data). Following [Cai and Lam, 2020] , we calculate the attention distribution over all input tokens:

$$\alpha_t = \text{softmax}(\frac{(h_{0,t}^v W^Q)(h_{1:n}^w W^K)^{\mathrm{T}}}{\sqrt{d}}), \tag{1}$$

where $\{W^Q, W^K\} \in R^{d\times d}$ are the weight matrices that project the graph state token and sentence tokens into key and query subspaces, respectively. For brevity, we simply mark the concatenation of $(h_1^w, \ldots, h_n^w)$ as $h_{1:n}^w \in R^{n\times d}$. The attention weight $\alpha_t \in R^n$ indicates the position of the node v_t coming from the input sentence S.

In addition, the node can also be generated from a pre-defined vocabulary. To calculate its probability distribution on the vocabulary, we firstly attend sentence tokens with α_t as the representation of the node $\alpha_t' \in R^d$. Then, we feed it into a fully-connected layer followed by a softmax classifier to produce the probability distribution $\mathcal{P}_{vocab,t}$:

$$\alpha_t' = \alpha_t h_{1:n}^w W^V, \tag{2}$$

$$\mathcal{P}_{vocab,t} = \text{softmax}(\alpha_t' W^{vocab}), \tag{3}$$

where $W^V \in R^{d\times d}$ denotes the parameter matrix that projects sentence tokens into the value subspace, and W^{vocab} denotes the weight matrix that transforms the dimension from d to the size of the vocabulary.

To control the contributions from the vocabulary and the input sentence, we utilize a soft switch layer to compute their weights:

$$[p_{0,t}, p_{1,t}] = \text{softmax}(\alpha_t' W^{switch}), \tag{4}$$

where W^{switch} is the weight matrix. We mark the contributions of vocabulary and input sentence as $p_{0,t}$ and $p_{1,t}$, respectively. Therefore, the final distribution of the next node $\mathcal{P}_t^v$ can be written as:

Algorithm 1 SIG generation

Input: Sentence $S = w_1, ..., w_n$
Output: SIG $G(V, E)$

$h_0^w, h_1^w, \ldots, h_n^w = Transformer(S)$
// Initialized SIG
$G_0 = (V = \{dummy\}, E = \{\emptyset\})$
// Begin interation graph spanning
for t=0 to MaxInteration **do**
  $h_{0,t}^v, \ldots, h_{t-1,t}^v = Transformer(G_t)$
  // o_t represents output layer of step t
  o_t^{node} = Node_Solver($h_{0:n}^w$,$h_{0,t:t-1,t}^v$)
  o_t^{edge} = Edge_Solver($h_{0:n}^w$,$h_{0,t:t-1,t}^v$)
  if o_t^{node} is end **then**
    break
  end if
  update G_t to G_{t+1}
end for
return G_t

$$\mathcal{P}_t^v = p_{0,t} \odot \mathcal{P}_{vocab,t} + p_{1,t} \odot (\sum_{j\in S(v_t)} \alpha_t[j]), \tag{5}$$

where $\odot$ denotes element-wise product, j is the index of tokens and $\alpha[j]$ is the j-th element of α_t. $S(v_t)$ is the index set of tokens corresponding to the node of graph.

Edge Solver. After obtaining the newly generated node v_t, the edge solver is used to decide the node's connection with previous nodes. To this end, the edge solver is designed to classify the edge types of nodes. We conduct multi-head attention between v_t and previous node states to calculate the attention weight over all existing nodes:

$$\beta_t^m = \text{softmax}(\frac{(h_{0,t}^v W_m^Q)(h_{0,t:t-1,t}^v W_m^K)^{\mathrm{T}}}{\sqrt{d}}), \tag{6}$$

where m represents the index of the head. Then, we take the maximum of attention weights over multiple heads as the final edge probability β_t. Edges whose probability value exceeds the threshold are kept [He and Choi, 2021]. Then we employ the deep biaffine classifier [Dozat and Manning, 2017] to assign their labels. The current graph node states $h_{1,t:t-1,t}^v$ are used to represent the dependencies in a biaffine decoder, which will generate the edge label score matrix $o_t^{edge\_label} \in R^{(t-1)\times k}$. Here k is the number of edge labels in the training data. The score matrix is used to predict whether the target node is the head of other nodes or not. Algorithm 1 details the generation processing of SIG.

2.3 Graph Aggregator

In this subsection, we describe our processing of detecting entities and triggers. To fully exploit the information of external knowledge between them, we propose a SIG aggregator to enrich the candidate nodes of entities and triggers to aggregate information from their neighbors based on the message passing mechanism. Previous study [Zhang and Ji, 2021] uses AMR graph to aggregate candidate nodes. However, the

noise in AMR lowers the extraction ability. Our SIG aggregator overcomes this by rebuilding the graph with the corresponding information. For example, the node in SIG will be replaced by the candidate node when meeting the same span.

Entity and Event Trigger Detection. This component aims to detect spans of entities and event triggers that could be used as candidate nodes. First, we feed sentence S into RoBERTa [Liu *et al.*, 2019] to obtain the contextual representation $X = [x_1, x_2, .., x_n]$. Afterwards, X is fed into conditional random field layers to determine the tags sequences of the entities and event triggers [Chiu and Nichols, 2016]. Let L^d be the log-likelihood of gold tag sequences for entities and triggers span detection. We obtain the entity spans $\{x_{start}, x_{end}\}$ and trigger spans $\{y_{start}, y_{end}\}$ by maximizing the log-likelihood L^d.

Graph Enriched. Once we have the SIG $G(V, E)$ and the candidate nodes set V^c, we map those candidate nodes to the SIG through corresponding spans in the original sentence. For each candidate node v_i^c in V^c, there are two situations in the map processing: (1) if v_i^c is already in $G(V, E)$ with the same span, we take the representation vector of the v_i^c as the initialized feature vector; (2) if v_i^c does not match any node span in the G, we add v_i^c into G as well as all related edges. Finally, we get the enriched graph $G^+ = (V^+, E^+)$.

GATs for Message Passing. Similar to [Zhang and Ji, 2021], we use GATs to propagate messages in the enriched graph G^+. Given the node $v_i^+ \in V^+$ and its connected neighbor set N_i, we compute the attention score $\alpha_{i,j}^l$ in l-th layer for each $v_j^+ \in N_i$ with the corresponding edge embedding $e_{i,j}$ by

$$\alpha_{i,j}^l = \frac{\exp(\sigma(f^l[W_n^l h_i^l : W_e^l e_{i,j} : W_n^l h_j^l]))}{\sum_{k \in N_i} \exp(\sigma(f^l[W_n^l h_i^l : W_e^l e_{i,k} : W_n^l h_k^l]))}, \quad (7)$$

where h_i^l is the l-th layer embedding of node v_i^+. W_n^l and W_e^l are trainable weight matrices for node and edge features. f^l denotes a fully connected layer. σ is an activation function, and we use LeakyReLU here.

We compute the message for node i by summing its neighbor features with weights:

$$m_i^l = \sum_{j \in N_i} \alpha_{i,j}^l h_i^l. \quad (8)$$

Finally, the updated node representation is calculated by

$$h_i^{l+1} = h_i^l + \gamma W_m^l m_i^l, \quad (9)$$

where γ controls the level of message passing between neighbors. W_m^l is the parameter. We select the last layer h^l as the final representation for each entity/trigger.

2.4 Training and Decoding

In this subsection, we introduce how we jointly decode the output information network with the identified entity, trigger nodes and their aggregated features h_i^l. We decide decoding order in two manners: a SIG-based top-down manner and a flat left to right manner.

Dataset	Split	Sents	Ents	Rels	Events
ACE05-E	Train	17,172	29,006	4,202	4,664
	Dev	923	2,451	450	560
	Test	832	3,017	403	636
ACE05-E+	Train	19,240	47,525	7,152	4,419
	Dev	902	3,422	728	468
	Test	676	3,673	802	424

Table 1: The statistics of ACE05-E and ACE05-E+

Optimal Graph with Global Constraint. The total score $S(G^{info})$ for the local information network is calculated by

$$S(G^{info}) = \sum_{t \in T} \sum_{i=1}^{N^t} FFN(h_i^l), \quad (10)$$

where h_i^l is the final representation of each entity or trigger node from the last layer of GATs. FFN is a feed-forward network. T is the set of IE subtasks, and N^t is the number of instances for task t. We further utilize global features f_g in [Lin *et al.*, 2020] and combine it with local score $S(G^{info})$ to compute the global score:

$$S^{'}(G^{info}) = S(G^{info}) + u^T f_g, \quad (11)$$

where f_g is a set of global feature templates to capture the cross-subtask and cross-instance interactions defined in [Lin *et al.*, 2020], and u is the weight vector of f_g. Finally, we maximize the following joint objective function

$$L = L^d + S(\hat{G}^{info}) - (S^{'}(\hat{G}^{info}) - S^{'}(G^{info})), \quad (12)$$

where $\hat{G}^{info}$ is the ground-truth network and G^{info} is the predicted information network.

Decoding Order Detection. Given all nodes and their pairwise edges, one way to output the information network G^{info} is to calculate the global score $S^{'}(G^{info})$ for each candidate graph. Then, we select the one with the highest global score in the beam search-based way [Lin *et al.*, 2020]. Besides, we also incorporate the SIG hierarchically to decide the top-down decoding manner [Zhang and Ji, 2021].

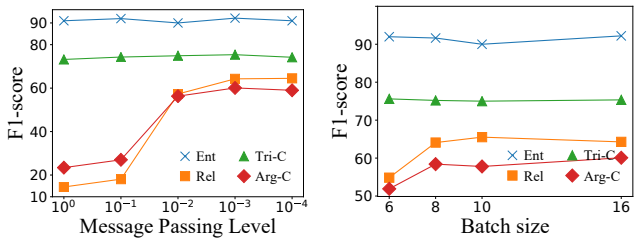
3 Experiments

3.1 Dataset

We conduct experiments on the Automatic Content Extraction (ACE) 2005 dataset, which provides the entity, relation and event annotations. Following the previous works of [Wadden *et al.*, 2019; Lin *et al.*, 2020], we use two different versions of ACE05 for the Joint IE task, *i.e.,* ACE05-E and ACE05-E+. **ACE05-E** is the widely used version of ACE05 dataset for event extraction. We follow the preprocessing method in [Wadden *et al.*, 2019]. **ACE05-E+** is proposed in [Lin *et al.*, 2020] by adding some important elements ignored by previous studies, which improves the quality and the data scale. Following these works, we keep 7 entity types, 6 relation types, 33 event types, and 22 event argument roles for ACE05-E and ACE05-E+. Table 1 shows the detailed statistics of ACE05-E and ACE05-E+.

Dataset	Model	Entity	Relation	Trg-I	Trg-C	Arg-I	Arg-C
ACE05-E	DyGIE++ [Wadden *et al.*, 2019]	89.7	-	-	69.7	53.0	48.8
	OneIE [Lin *et al.*, 2020]	90.2	-	78.2	74.7	59.2	56.8
	FourIE [Nguyen *et al.*, 2021]	91.3	-	78.3	**75.4**	60.7	58.0
	AMRIE [Zhang and Ji, 2021]	92.1	62.3	78.1	75.0	60.9	58.6
	InterIE	**92.2**	**63.9**	78.2	75.3	**61.4**	**59.4**
	InterIE w/o Dec	**92.2**	63.5	**79.1**	75.2	59.7	57.7
ACE05-E+	OneIE [Lin *et al.*, 2020]	89.6	58.6	75.6	72.8	57.3	54.8
	FourIE [Nguyen *et al.*, 2021]	91.1	63.6	76.7	73.3	59.5	57.5
	AMRIE [Zhang and Ji, 2021]	92.2	62.6	78.5	75.2	61.2	58.5
	InterIE	92.2	64.3	**78.8**	**75.3**	**62.3**	**60.1**
	InterIE w/o Dec	**92.4**	**65.5**	78.0	74.9	60.6	59.4

Table 2: Overall F1-scores (%) of joint information extraction. InterIE w/o Dec is model ablation variants where we only keep the left to right decoding order. "-" indicates results are not reported in their work and AMRIE on ACE05-E+ performance is our reproduction.



(a) Effect of Message Passing γ (b) Effect of Batch Size

Figure 3: Performace on ACE05-E+ dataset changes with paramenters.

3.2 Experimental Setup

We use the same evaluation metrics (*i.e.,* F1) with the previous works. The criteria for each subtask are as follows.

Entity. Both the offsets and type of entities are correct.

Relation. The entities of subject and object and the relation type are correct.

Event Trigger. A trigger (Trg-I) is identified correctly if its offsets are correct. A trigger (Trg-C) is classified correctly if its event type is also correct.

Argument. An argument (Arg-I) is correctly identified if its event type and offsets are correct. An argument (Arg-C) is correctly classified if its role type is also correct.

We train our model with Adam [Kingma and Ba, 2015] on NVIDIA 3090 with a learning rate 2e-5 for RoBERTa parameters and 4e-4 for others. The number of epoch is 100. To get a fair comparison, we use the same settings of other parameters with [Lin *et al.*, 2020; Zhang and Ji, 2021]. Our code is released to support research[1].

3.3 Overall Performance

The two datasets have been used widely in IE. This allows us directly compare the results of our InterIE against baselines in the same experimental setting.

[1]https://github.com/LucyFann/InterIE

Baseline Methods. We adopt the most recent joint IE models as our baselines. **DyGIE++** [Wadden *et al.*, 2019] is a joint IE framework using local-specific classifier with span representations. **OneIE** [Lin *et al.*, 2020] is a graph-based joint IE framework decoding with global features. **AMRIE** [Zhang and Ji, 2021] uses AMR graph to guide encoding and decoding based on OneIE. **FourIE** [Nguyen *et al.*, 2021] applies GNN to perform type predictions and regularizations.

Table 2 lists the F1 of baseline models on the two datasets reported in their papers. Our InterIE is listed in the last row of each dataset. We have the following observations from the experiment results:

- InterIE achieves improvements almost on all IE subtasks, including entity, relation and event extractions on both ACE05-E and ACE05-E+ datasets. The performance greatly outperforms on edge classification tasks such as relation extraction ($\uparrow$ 2.9% and $\uparrow$ 1.6%) and event argument role labeling ($\uparrow$ 1.6% and $\uparrow$ 0.8%). This indicates InterIE can better recognize relations between entities and event triggers with the incorporation of SIG.
- InterIE performs better on ACE05-E+ than ACE05-E. This is because ACE05-E+ adds back the ignored information, including the order of relation arguments, pronouns, and multi-token event triggers. It tells us that the InterIE model is good at handling multi-event and multi-relationship on sentence level, which demonstrates the rooted reentrance structure of SIG captures the interdependency of different IE subtasks.
- To further show the influence of decoder order, two decoding orders are compared. In InterIE w/o Dec, we keep the flat left-to-right decoding order, while InterIE uses hierarchical order based on SIG. From the results, we can see that InterIE is not sensitive to the decoder order. It indicates that our model breakthroughs the limitations of AMRIE.

3.4 Analysis on Hyperparameters

We conduct hyperparameters analysis including γ defined in 8 and batch size. Figure 3(a) shows the effect of γ, which ranges from 10^{-4} to 1. The result shows that when γ contin-

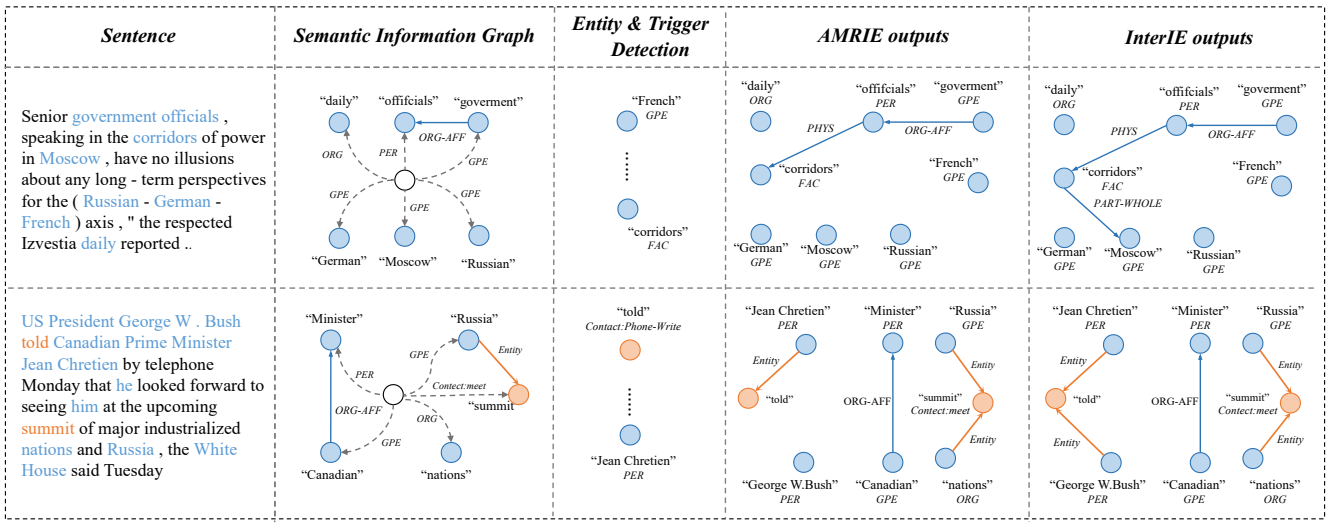


Figure 4: Case study illustrates how SIG improves the performance of joint IE comparing with AMRIE outputs

ually increases higher than 10^{-2}, the performance of the subtasks will decrease rapidly. This draws a similar conclusion to [Zhang and Ji, 2021]. It makes sense because if the nodes focus too much on their neighborhood information, they will lose some of their inherent semantic features, which causes a significant performance decrease. Figure 3(b) shows the performance of our InterIE model improves significantly with a larger beam size. When the batch size reaches 8, the model performance reaches a smooth level. In addition, we find that mapping all IE edge types from SIG to GATs for the training process may lead to overfitting, so we only selected edge types that represent relation extraction and events.

3.5 Case Study

To give an intuitive understanding, we select two sentences from the testing datasets to illustrate how the SIG jointly helps the relation and event extractions. Figure 4 shows the complementation mechanism. From the first sentence in Figure 4, we find that the SIG and recognized entities (*i.e., French*) and triggers could be complemented to improve relation extraction. On the other hand, SIG keeps the relation of unrecognized entities and triggers, *e.g.,* the "ORG-AFF" between *official* and *government*. In summary, SIG could enhance the recognized elements and keep the unrecognized elements to improve the extraction ability. From the second example in Figure 4, we observe that SIG detects the trigger (*i.e., summit*), which is not recognized by the entity and trigger detection. Meanwhile, entity and trigger detection recognize the "nations", which is the argument role of the event. From the two above examples, we find that the well-designed SIG and entity/trigger detection is complementary to achieve a synergistic effect.

4 Related Work

Early works solve the joint information extraction subtasks with feature engineering such as Lexical and WordNet features [Ahn, 2006], global constraints [Roth and Yih, 2004; Li *et al.*, 2014], Markov Logic Networks [Venugopal *et al.*, 2014; Poon and Domingos, 2007]. However, these studies still fail to consider all of the three main IE subtasks together in a joint way. Since the IE subtasks can be considered as the problem of building an Information network $G^{info}(V, E)$ [Li *et al.*, 2014], graph-based methods have been proposed to do subtasks jointly. With the development of deep learning, [Wadden *et al.*, 2019] designed a neural model using contextualized embeddings in local-specific classifiers for different IE subtasks. [Lin *et al.*, 2020] exploits BERT and global features to constrain the decoding step. Meanwhile, some works focus on entity and relation extraction [Zheng *et al.*, 2017; Zhong and Chen, 2020; Fu *et al.*, 2019] and event extraction [Zhang *et al.*, 2018; Liu *et al.*, 2018; Zhang *et al.*, 2021], respectively. These works can not scale to extracting a information network of all IE subtasks. With the help of graph neural networks(GNN), some studies take further steps to capture the interaction among IE subtasks. [Nguyen *et al.*, 2021] applies GNN to perform type predictions and regularizations. [Zhang and Ji, 2021] introduce AMR graph to guide the encoding and decoding stages for IE subtasks.

5 Conclusion

In this study, we propose InterIE model based on SIG for jointly extracting information (*i.e.,* entity, relation, and event). The key idea of InterIE is to design a SIG aggregator to capture the inter-dependency between different IE subtasks. The major difference with AMRIE is that AMR graph introduces redundant semantic noise, which leads to redundant interference during the encode and decode stages. To this end, we design the rebuilding mechanism for SIG to avoid the noise inherent in AMR. Experiment results show that the proposed InterIE achieves state-of-the-art performance on all subtasks of IE. More importantly, InterIE is not sensitive to decoding order, which shows the robustness of our InterIE model.

Acknowledgments

This work is supported by the National Key R&D Program of China (2020AAA0105200) and the National Science Foundation of China (NSFC No. 62106249, U2001212, 62032001, and 61932004).

References

[Ahn, 2006] David Ahn. The stages of event extraction. In *Proceedings of the Workshop on Annotating and Reasoning about Time and Events*, pages 1–8, Sydney, Australia, July 2006. Association for Computational Linguistics.

[Banarescu *et al.*, 2013] Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. Abstract meaning representation for sembanking. In *ACL*, pages 178–186, 2013.

[Cai and Lam, 2020] Deng Cai and Wai Lam. AMR parsing via graph-sequence iterative inference. In *ACL*, pages 1290–1301, 2020.

[Chiu and Nichols, 2016] Jason P. C. Chiu and Eric Nichols. Named entity recognition with bidirectional lstm-cnns. *Trans. Assoc. Comput. Linguistics*, 4:357–370, 2016.

[Dozat and Manning, 2017] Timothy Dozat and Christopher D. Manning. Deep biaffine attention for neural dependency parsing. In *ICLR*, 2017.

[Fu *et al.*, 2019] Tsu-Jui Fu, Peng-Hsuan Li, and Wei-Yun Ma. Graphrel: Modeling text as relational graphs for joint entity and relation extraction. In *ACL*, pages 1409–1418, 2019.

[He and Choi, 2021] Han He and Jinho D. Choi. Levi graph AMR parser using heterogeneous attention. *CoRR*, abs/2107.04152, 2021.

[Kingma and Ba, 2015] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

[Li *et al.*, 2014] Qi Li, Heng Ji, Yu Hong, and Sujian Li. Constructing information networks using one single model. In *ACL*, pages 1846–1851, 2014.

[Lin *et al.*, 2020] Ying Lin, Heng Ji, Fei Huang, and Lingfei Wu. A joint neural model for information extraction with global features. In *ACL*, pages 7999–8009, 2020.

[Liu *et al.*, 2018] Xiao Liu, Zhunchen Luo, and Heyan Huang. Jointly multiple events extraction via attention-based graph information aggregation. In *EMNLP*, pages 1247–1256, 2018.

[Liu *et al.*, 2019] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692, 2019.

[Luan *et al.*, 2019] Yi Luan, Dave Wadden, Luheng He, Amy Shah, Mari Ostendorf, and Hannaneh Hajishirzi. A general framework for information extraction using dynamic span graphs. In *NAACL-HLT*, pages 3036–3046, 2019.

[Nguyen *et al.*, 2021] Minh Van Nguyen, Viet Lai, and Thien Huu Nguyen. Cross-task instance representation interactions and label dependencies for joint information extraction with graph convolutional networks. In *NAACL-HLT*, pages 27–38, 2021.

[Poon and Domingos, 2007] Hoifung Poon and Pedro M. Domingos. Joint inference in information extraction. In *AAAI*, pages 913–918, 2007.

[Roth and Yih, 2004] Dan Roth and Wen-tau Yih. A linear programming formulation for global inference in natural language tasks. In *HLT-NAACL*, pages 1–8, 2004.

[Velickovic *et al.*, 2018] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *ICLR*, 2018.

[Venugopal *et al.*, 2014] Deepak Venugopal, Chen Chen, Vibhav Gogate, and Vincent Ng. Relieving the computational bottleneck: Joint inference for event extraction with high-dimensional features. In *EMNLP*, pages 831–843, 2014.

[Wadden *et al.*, 2019] David Wadden, Ulme Wennberg, Yi Luan, and Hannaneh Hajishirzi. Entity, relation, and event extraction with contextualized span representations. In *EMNLP-IJCNLP*, pages 5783–5788, 2019.

[Walker and Consortium, 2005] C. Walker and Linguistic Data Consortium. *ACE 2005 Multilingual Training Corpus*. LDC corpora. Linguistic Data Consortium, 2005.

[Zhang and Ji, 2021] Zixuan Zhang and Heng Ji. Abstract meaning representation guided graph encoding and decoding for joint information extraction. In *NAACL-HLT*, pages 39–49, 2021.

[Zhang *et al.*, 2018] Yuhao Zhang, Peng Qi, and Christopher D. Manning. Graph convolution over pruned dependency trees improves relation extraction. In *EMNLP*, pages 2205–2215, 2018.

[Zhang *et al.*, 2021] Junchi Zhang, Qi He, and Yue Zhang. Syntax grounded graph convolutional network for joint entity and event extraction. *Neurocomputing*, 422:118–128, 2021.

[Zheng *et al.*, 2017] Suncong Zheng, Feng Wang, Hongyun Bao, Yuexing Hao, Peng Zhou, and Bo Xu. Joint extraction of entities and relations based on a novel tagging scheme. In *ACL*, pages 1227–1236, 2017.

[Zhong and Chen, 2020] Zexuan Zhong and Danqi Chen. A frustratingly easy approach for joint entity and relation extraction. *CoRR*, abs/2010.12812, 2020.