Constrained Tuple Extraction with Interaction-Aware Network

Xiaojun Xue, Chunxia Zhang*, Tianxiang Xu, Zhendong Niu
School of Computer Science and Technology, Beijing Institute of Technology, Beijing, China
{xiaojunx, cxzhang, txxu, zniu}@bit.edu.cn

Abstract

Tuples extraction is a fundamental task for information extraction and knowledge graph construction. The extracted tuples are usually represented as knowledge triples consisting of subject, relation, and object. In practice, however, the validity of knowledge triples is associated with and changes with the spatial, temporal, or other kinds of constraints. Motivated by this observation, this paper proposes a constrained tuple extraction (CTE) task to guarantee the validity of knowledge tuples. Formally, the CTE task is to extract constrained tuples from unstructured text, which adds constraints to conventional triples. To this end, we propose an interaction-aware network. Combinatorial interactions among context-specific external features and distinct-granularity internal features are exploited to effectively mine the potential constraints. Moreover, we have built a new dataset containing totally 1,748,826 constrained tuples for training and 3656 ones for evaluation. Experiments on our dataset and the public CaRB dataset demonstrate the superiority of the proposed model. The constructed dataset and the codes are publicly available.[1]

1 Introduction

Tuples extraction task aims to extract knowledge tuples from unstructured texts, which is a fundamental task for information extraction, knowledge graph construction, and so on (Cui et al., 2018; Jiang et al., 2019; Banerjee and Baral, 2020; Li et al., 2022). The extracted knowledge tuples are mainly represented in the form of (subject, relation, object) (Jiang et al., 2019), and are usually acquired through named entity recognition (NER), relation extraction (RE) (Zhong and Chen, 2021; Li et al., 2020; Yahya et al., 2014), and open information extraction (Open IE) (Jiang et al., 2019; Wang et al., 2022). With the rise of large-scale pretraining methods (e.g. BERT (Devlin et al., 2019)), the quality of knowledge triples has improved significantly.

However, current knowledge triples lack constraints for their authenticity. In practice, constraints are ubiquitous in numerous domains, such as spatial, temporal, conditional and environmental ones. Constraints are essential supplements to knowledge triples, and play an instructional role. Our deep investigation on literature indicates that this topic has not gained enough attention.

Some works utilize temporal knowledge tuples to reflect temporal dynamics (Gracious et al., 2021; Jung et al., 2021). However, only temporal constraints are not sufficient and general enough to guarantee the validity of knowledge triples. For instance, let us consider the following sentence:

"Consuming the same power, the performance of ARM CPU is better than that of Intel CPU.", only if the conditional constraint *"consuming the same power"* is satisfied, the knowledge triple "*(ARM CPU, better than, Intel CPU)*" is true. The above observations motivate us to add constraints to conventional knowledge triples for the validity of knowledge tuples. Thereby, we propose a novel task called constrained tuple extraction (CTE), which aims to provide knowledge tuples with temporal, spatial, conditional constraints in general domain. CTE represents knowledge tuples in the form of (subject, relation, object, constraint). Constraints are some phrase descriptions that guarantee the validity of knowledge tuples, so that the knowledge tuples can be effectively utilized.

The constrained tuples are extracted via Open IE approaches, similar to the works in Jiang et al. (2019); Wang et al. (2022). This extraction approach is more general and doesn't require predefined entity and relation types. Table 1 shows some examples for CTE task. It is worth noting that the constrained tuple extraction task aims at providing an explicit and uniform information extraction technique to guarantee the validity of knowledge

*Corresponding author.

[1]https://github.com/ckgconstruction/ckg.

Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 11430–11444

Table 1: Constrained knowledge tuples extracted from sentences in CTE task.

Constraints	Sentences	Tuples
Conditional	***Consuming the same power***, *the performance of ARM CPU is better than that of Intel CPU.*	*(ARM CPU, better than, Intel CPU,* ***consuming the same power****)*
Spatial	*British troops formally shut down their base* ***in Helmand****.*	*(British troops, shut down, their base,* ***in Helmand****)*
Temporal	*Obama served as the 44th president of the United States* ***from 2009 to 2017****.*	*(Obama, the 44th president of, the United States,* ***from 2009 to 2017****)*

(or information) tuples, rather than a new form of knowledge representation.

In addition, an interaction-aware network is developed to fulfill CTE task from unstructured texts. We argue that constraints in knowledge tuples affect the validity and are contained in deep-level conceptual semantics rather than shallow contexts. Interactions among multiple textual features can further mine implicit semantic information beyond context, which is beneficial to discovering potential constraints.

Architecturally, the interaction-aware network consists of three modules: context-specific enhancement module, distinct-granularity feature extraction module and interaction-aware module. Specifically, the first module is designed to learn the context-specific external features by leveraging multi-view graphs. The second module is to sufficiently extract the distinct-granularity (e.g. phrase-level, word-level, and global-level) internal context features. The third module is developed to achieve the goal of interaction between the external and internal features in a combination way utilizing distribution-sparse multi-head attention. The combinatorial interactions occur between any two external or internal features. Thus, this trick yields a mechanism in that the deep-level conceptual semantic can be explored to help mine the potential constraints existing in knowledge.

The contributions of this paper are summarized as follows:

- We propose a novel task called constrained tuple extraction (CTE), whose mission is to extract knowledge tuples with temporal, spatial, conditional constraints. A new manually annotated Constrained Tuple Extraction Benchmark (CTEB) dataset for the CTE task is built and publicly available.
- Interaction-aware network (IAN) is proposed to fulfill CTE task, which facilitates the combinatorial interactions among the context-specific external features and the distinct-granularity internal features to effectively mine the potential constraints in knowledge.
- Distribution-sparse multi-head attention is designed not only to select the dominating attentions but also to facilitate efficient interactions.

2 Related Work

The goal of the tuples extraction task is to extract knowledge tuples from unstructured texts (Jiang et al., 2019; Banerjee and Baral, 2020; Li et al., 2022). The extracted knowledge tuples are mainly acquired through relational triple extraction (Zhong and Chen, 2021; Li et al., 2020; Yahya et al., 2014) and open information extraction (Jiang et al., 2019; Wang et al., 2022).

2.1 Open Information Extraction

Open information extraction (Open IE) aims to extract predicates and corresponding arguments from unstructured texts in open domain, without predefining entity and relation types. The extracted predicates and corresponding arguments can constitute knowledge tuples. Open IE methods mainly include rule-based ones (Fader et al., 2011; Corro and Gemulla, 2013; Angeli et al., 2015) and neural network based ones (Stanovsky et al.; Cui et al., 2018; Wang et al., 2022). The neural network models are further divided into sequence labeling ones (Stanovsky et al.; Roy et al., 2019; Jiang et al., 2020), sequence generation ones (Cui et al., 2018; Sun et al., 2018; Kolluru et al., 2020b) and span-based ones (Zhan and Zhao, 2020).

Specifically, Ro et al. (2020) proposed Multi2OIE to utilize BERT to extract predicates.

Then the BERT hidden feature, position embedding and predicate average feature were input into multi-head attention blocks to extract arguments. Solawetz and Larson (2021) proposed SRL_BERT to improved RnnOIE (Stanovsky et al., 2018) by replacing the bidirectional encoder with BERT and the predicate index embedding with sentence embedding. Jiang et al. (2019) proposed a three-layer structure for scientific tuple extraction using Open IE methods. The tuples were divided into fact tuples and condition tuples. In addition, OpenIE6 (Kolluru et al., 2020a) utilized two-dimensional grid labeling to improve the extraction efficiency for Open IE task. Later, DetIE (Vasilkovsky et al., 2022) regarded the tuples as three-dimensional anchor boxes, and improved the extraction speed by that single-shot approach. Tuples extraction based on Open IE has better generality, because it does not require predefined entity and relation types. Open IE task focuses on the structure of predicates and corresponding arguments, while CTE task focuses on providing a more unified form to ensure the validity of knowledge tuples.

2.2 Relational Triple Extraction

Relational triple extraction is mainly accomplished by named entity recognition (NER) and relation extraction (RE) (Fu et al., 2020; Zhong and Chen, 2021).

Typically, Li et al. (2020) extracted relational triples from free texts in the e-commerce field based on NER and RE. Moreover, Zhong and Chen (2021) explicitly injected positions and categories information of entities into the input sentences for relation extraction, so that different contextual representations were learned for entities and relations. Nevertheless, relational triple extraction based on NER and RE is usually limited to the domain portability.

3 Methodology

3.1 Problem Definition

Given a piece of text m, the constrained tuple extraction task is to extract n_t (>=1) constrained knowledge tuples formatted as t = (subject, relation, object, constraint) from each sentence in text m. The constraints can be formatted as temporal expressions, spatial descriptions, and conditional forms. Formally, the constrained knowledge tuples t can be formulated as:

$$t = (s, r, o, c), \tag{1}$$

where s, r, o, c represent the subject, relation, object and constraint, respectively.

3.2 Overview of the Proposed Model

Figure 1 shows the architecture of the proposed interaction-aware network (IAN) for CTE task. IAN consists of three modules: context-specific enhancement module, distinct-granularity feature extraction module, and interaction-aware module. First, context-specific enhancement module is developed to leverage multi-view graphs and learn context-specific external features for input text. Second, to sufficiently exploit the inherent features of the text, distinct-granularity internal features are explicitly extracted from the raw text. Finally, interaction-aware module is designed to facilitate the combinatorial interactions between any two features in the context-specific external and the distinct-granularity internal features. Meanwhile, distribution-sparse multi-head attention is proposed to select the dominating attentions and alleviate the interaction deficiency problem.

3.3 Context-Specific Enhancement Module

External auxiliary features can provide additional semantic information for constrained tuple extraction. The existing methods usually introduce entity-specific external knowledge. The introduced external features only consider the entity itself rather than the context of the entity. It results in that the introduced external features are indistinguishable for the same entities in different texts (Li et al., 2020). In this paper, different from them, context-specific external features are introduced for spans in the text according to the contexts of spans. Thus, the introduced external features can provide context-specific auxiliary information and enrich raw texts.

In this work, Wikidata[2] is used to leverage and generate preliminary auxiliary information. First, the candidate spans in the input text are matched and aligned with the data in Wikidata. Thus, the potential entity nodes in Wikidata corresponding to the candidate spans can be obtained. Then, centered on the potential entity nodes, their two-hop neighbors and edges in Wikidata are regarded as preliminary auxiliary information. Therefore, the two-hop graph structures $\mathbf{G}$ are generated according to the aligned spans and Wikidata.

To generate context-specific external features, multi-view graphs are generated from $\mathbf{G}$ for each

[2] https://www.wikidata.org/wiki/Wikidata

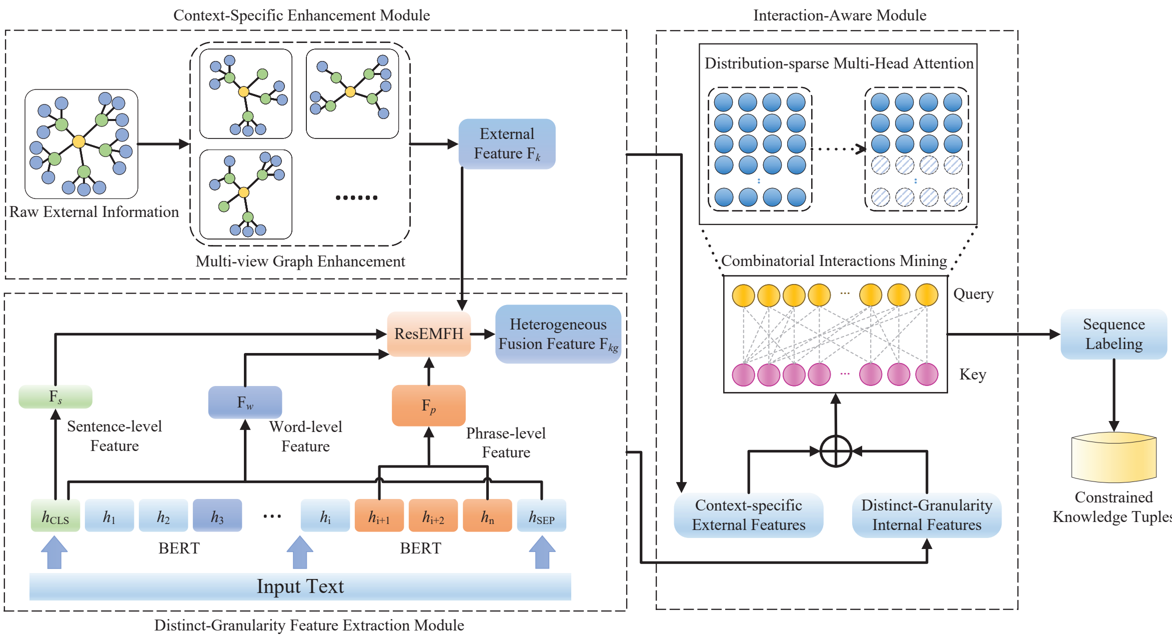


Figure 1: Overview of interaction-aware network (IAN) for constrained tuple extraction (CTE).

aligned span. Inspired by the works in Nathani et al. (2019) and Xue et al. (2021), the auxiliary information is represented in the form of graph, which contains nodes (potential entities) and edges (attributes or relations) in Wikidata related to the aligned spans. Specifically, both sentence and the contents of the nodes in the related two-hop graph structure are input into BERT (Devlin et al., 2019). For each aligned span in each sentence, the initial external auxiliary information is defined as $\mathbf{V}^0 = \{v_1^0, v_2^0, ..., v_{T+1}^0\}$. Each element in $\mathbf{V}^0$ corresponds to a node in $\mathbf{G}$. Gaussian graph generator is used to generate the potential multi-view graphs. The contextual feature is captured when encoding each node into Gaussian distributions:

$$\{\mu_i^1, \mu_i^2, ..., \mu_i^N\} = f_\theta(v_i^0, h_{\text{CLS}}), \tag{2}$$

$$\{\sigma_i^1, \sigma_i^2, ..., \sigma_i^N\} = \phi(f_\theta^{'}(v_i^0, h_{\text{CLS}})), \tag{3}$$

where θ denotes SoftPlus activation function, h_{CLS} is the representation of CLS token, f_θ and $f_\theta^{'}$ represent two learnable neural networks, N is the number of views. Then a series of Gaussian distributions $\{\delta_1^n, \delta_2^n, ..., \delta_{T+1}^n\}$ are obtained for the n-th view in the multi-view graph.

Next, Kullback-Leibler (KL) divergence between the Gaussian distributions of two nodes is used to generate weights for edges. In addition, due to the asymmetry of KL divergence, the obtained multi-view graph is a directed graph. The edge weight between two nodes in the n-th view is calculated as:

$$e_{ij}^n = \text{KL}(\delta_i^n(\mu_i^n, \sigma_i^{n^2})||\delta_j^n(\mu_j^n, \sigma_j^{n^2})). \tag{4}$$

Totally, N adjacency matrices $\mathbf{M}^1, \mathbf{M}^2, ..., \mathbf{M}^N$ are acquired after getting the weights of edges between nodes. Hence, we can get the multi-view graphs $\mathbf{G}_m = \{\mathbf{V}^0, \mathbf{M}^1, \mathbf{M}^2, ..., \mathbf{M}^N\}$.

Next, the embedding generation method KBGAT (Nathani et al., 2019) is used to generate embeddings for the nodes in $\mathbf{G}_m$ (i.e. the aligned spans in text). The KBGAT method considers both nodes and edges features to generate embeddings for graphs, and uses *hinge*-loss as the training objective. The inputs of KBGAT are node embeddings matrix $\mathbf{E} \in \mathbb{R}^{l_e \times d_e}$ and edge embeddings matrix $\mathbf{R} \in \mathbb{R}^{l_r \times d_r}$. Here, l_e and l_r represent the total numbers of nodes and edges respectively, d_e denotes the dimension of each node embedding, d_r indicates the dimension of each edge embedding. The adjacency matrices in KBGAT are replaced by $\mathbf{G}_m$ to generate embeddings for the nodes in $\mathbf{G}_m$. Then, context-specific external features $\mathbf{F}_k$ can be obtained for aligned spans in text. More details about $\mathbf{F}_k$ generation are shown in Appendix A.3.

3.4 Distinct-Granularity Feature Extraction Module

Aiming to sufficiently exploit the inherent textual features, distinct-granularity internal features are extracted from the raw text.

Since subjects, objects, relations and constraints usually appear in form of phrases in real world, phrase-level features play an important role in constrained tuple extraction. AutoPhrase (Shang et al., 2018) is employed to explicitly mine the phrases in the text. To generate contextualized phrase features, we insert phrase start tag $<\text{PHRASE}>$ and phrase end tag $</\text{PHRASE}>$ into sentences. Then, the BERT (Devlin et al., 2019) embeddings of the phrase tags and the words in phrases are averaged to obtain the phrase-level feature $\mathbf{F}_p$.

$$\mathbf{F}_p = \text{BERT}(AutoPhrase(m)). \tag{5}$$

Then, BERT is utilized to generate embeddings for words and sentences. The hidden states of last four layers of BERT are added to generate word-level feature $\mathbf{F}_w$. The sentence-level feature $\mathbf{F}_s$ is represented by the $[\text{CLS}]$ representation in the hidden states of last layer of BERT. The representations in the last layer of BERT are denoted as BERT hidden $\mathbf{F}_r$.

In order to enable the internal and external features to interact directly with the text global feature, the phrase-level, sentence-level, word-level and external features are fused to obtain the textual heterogeneous fused feature $\mathbf{F}_g$. By doing so, $\mathbf{F}_g$ is taken as a separate feature and can be utilized to mine the combinatorial interactions in the interaction-aware module. By using the multimodal fusion method EMFH (Xue et al., 2023; Yu et al., 2018), different from them, residual connections are added among the multiple EMFB blocks to fuse $\mathbf{F}_p, \mathbf{F}_s, \mathbf{F}_w$ and the context-specific external feature $\mathbf{F}_k$. We denote the improved fusion method as ResEMFH. The EMFB block is calculated as:

$$z_e^i = DP(\varphi(\tilde{\mathrm{U}}_{\mathrm{k}}^{\mathrm{T}}\mathbf{F}_k \circ \tilde{\mathrm{U}}_{\mathrm{w}}^{\mathrm{T}}\mathbf{F}_w \circ \tilde{\mathrm{U}}_{\mathrm{s}}^{\mathrm{T}}\mathbf{F}_s \circ \tilde{\mathrm{U}}_{\mathrm{p}}^{\mathrm{T}}\mathbf{F}_p)), \tag{6}$$

$$z_q^i = Norm(SumPool(z_e)), \tag{7}$$

where φ denotes the $tanh$ activation function, DP represents dropout operation, z_e^i and z_q^i stand for the outputs of expand stage and squeeze stage in the i-th EMFB block, respectively.

Next, L_g EMFB blocks are then cascaded via residual connections. Finally, the outputs of L_g EMFB blocks are averaged to acquire textual heterogeneous fused feature $\mathbf{F}_g$:

$$z_q^{i+1} = z_q^i + EMFB(z_e^i, \mathbf{F}_k, \mathbf{F}_w, \mathbf{F}_s, \mathbf{F}_p), \tag{8}$$

$$\mathbf{F}_g = Mean(z_q^1, z_q^2, ..., z_q^{L_g}). \tag{9}$$

To provide predicate mentions for constrained tuple extraction, we utilize predicate feature and position embedding used in the work of Ro et al. (2020). Here, predicate feature is obtained by averaging and duplicating the hidden states of the predicates. Position embedding uses binary values to represent the positions of predicate spans. Finally, word-level, sentence-level, phrase-level textual features $\mathbf{F}_w$, $\mathbf{F}_s$, $\mathbf{F}_p$, BERT hidden $\mathbf{F}_r$, heterogeneous fused feature $\mathbf{F}_g$, predicate feature and position embedding together constitute distinct-granularity internal features.

3.5 Interaction-Aware Module

The purpose of the interaction-aware module is to mine the combinatorial interactions between any two features in context-specific external features and distinct-granularity internal features. These combinatorial interactions contain implicit semantics and deep-level correlations besides the contextual information. Previous works usually use multi-head attention for information extraction (Ro et al., 2020; Vaswani et al., 2017). However, we observe that the self-attention used in multi-head attention suffers from interaction deficiency and attention sparsity problems. That is, interactions occur only among minority specific features, rather than among all external and internal features, as illustrated in Section 4.4. Meanwhile, a part of query key pairs dominate the main attention weights, and there are many non-key attention weights.

For above purposes, we propose distribution-sparse multi-head attention mechanism, which selects the dominating attentions in each head, and facilitates the interactions among context-specific external features and distinct-granularity internal features. It is worth noting that only the selected dominating query-key pairs are calculated.

The general self-attention takes query $\mathbf{Q}$, key $\mathbf{K}$ and value $\mathbf{V}$ as inputs. In this module, context-specific external and distinct-granularity internal features are concatenated to form the feature $\mathbf{F}_{arg}$, which is utilized to generate key-value pairs and query. $\mathbf{F}_{arg}$ itself is regarded as a query. Key-value pairs are subsets of $\mathbf{F}_{arg}$ derived from the predicate positions. Let q_i, k_i, v_i denote the i-th row in $\mathbf{Q}$, $\mathbf{K}$, $\mathbf{V}$, respectively. L_Q, L_K, L_V represent the numbers of rows in $\mathbf{Q}$, $\mathbf{K}$, $\mathbf{V}$. Following previous works (Tsai et al., 2019; Zhou et al., 2021), the attention of the i-th query is calculated as:

$$\text{Attention}(q_i, \mathbf{K}, \mathbf{V}) = \sum_j \frac{k(q_i, k_j)}{\sum_l k(q_i, k_l)} v_j, \tag{10}$$

$$p(k_j|q_i) = \frac{k(q_i, k_j)}{\sum_l k(q_i, k_l)}, \tag{11}$$

where $k(q_i, k_j)$ selects the exponential kernel $\exp(q_i k_j^\top / \sqrt{d})$, d represents the input dimension. The quadratic dot-product computation is required in self-attention, and the memory usage scales in $O(L_Q L_K)$. This is a bottleneck to enhance the information extraction capacity.

It can be observed from Eq.(10) and Eq.(11) that the attention weight of the i-th query is obtained by calculating the compatibility $p(k_j|q_i)$. The output is a combination of attention weights and values v. The dominating query-key pairs encourage the attention distribution away from the mean distribution $q(k_j|q_i) = 1/L_K$. Inspired by Zhou et al. (2021), by measuring the "difference" between distribution p and distribution q, important query-key pairs can be distinguished. Using dominating query-key pairs can filter out redundant interactions and allow the model to focus on effective combinatorial interactions. Meanwhile, the attention sparsity problem can be alleviated, and the computation and memory usage can be reduced. For this purpose, Kullback-Leibler divergence is utilized to measure the "difference". The sparsity measurement for the i-th query is formulated as:

$$S(q_i, \mathbf{K}) = \ln \sum_{j=1}^{L_K} e^{\frac{q_i k_j^\top}{\sqrt{d}}} - \frac{1}{L_K} \sum_{j=1}^{L_K} \frac{q_i k_j^\top}{\sqrt{d}}, \tag{12}$$

where the two terms in Eq.(12) are the Log-Sum-Exp and the arithmetic mean of q_i on all the keys, respectively. The distribution probability p is more diverse and more likely to include important query-key pairs when $S(q_i, \mathbf{K})$ for the i-th query is larger. To further reduce the calculation for traversal of queries when computing the sparsity measurement, according to Calafiore et al. (2020), the sparsity measurement can be empirically approximated as:

$$\widetilde{S}(q_i, \mathbf{K}) = \underset{j}{max}\{\frac{q_i k_j^T}{\sqrt{d}}\} - \frac{1}{L_K} \sum_{j=1}^{L_K} \frac{q_i k_j^T}{\sqrt{d}}. \tag{13}$$

Subsequently, only $L_K ln L_Q$ dot-product pairs are randomly sampled to calculate the $\widetilde{S}(q_i, \mathbf{K})$. The other pairs are filled with zero. According to the above sparsity measurement, distribution-sparse attention is designed to make each key focus only on Top-h dominating queries:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}(\frac{\widetilde{\mathbf{Q}}\mathbf{K}^\top}{\sqrt{d}})\mathbf{V}, \tag{14}$$

where $\widetilde{\mathbf{Q}}$ is a sparse matrix and contains only the sparse Top-h queries. A constant sampling factor c is utilized to control $h = c * lnL_Q$. Thus, for each query-key lookup, the distribution-sparse self-attention calculates $O(lnL_Q)$ dot-product. Meanwhile, the memory usage is $O(L_K lnL_Q)$. Different sparse query-key pairs are generated and calculated in each head. Then sparse Top-h from them are selected. Here, queries and keys has the same shape, thus time complexity and space complexity of the distribution-sparse self-attention are reduced to $O(NlnN)$.

The CTE task is regarded as a sequence labeling task. Similar to the previous work Multi^{2}OIE(Ro et al., 2020), two-stage extraction strategy is adopted. Predicates are first identified, and then subjects, objects, and constraints are extracted. In the first stage, BERT hidden $\mathbf{F}_r$ is input into a fully connected (FC) layer to classify predicates. In the second stage, the context-specific external features $\mathbf{F}_k$ and the distinct-granularity internal features are concatenated to form feature $\mathbf{F}_{arg}$. Then, $\mathbf{F}_{arg}$ is fed into the distribution-sparse multi-head attention followed by another FC layer to obtain the final constrained knowledge tuples. Cross-entropy loss is used to train the IAN model, same as that in Multi^{2}OIE.

4 Experiments

4.1 Experimental Setup

Datasets: To verify the constrained tuple extraction task and the interaction-aware network, two public datasets are utilized in this section. The first dataset is a new manually annotated dataset called **Constraint Tuples Extraction Benchmark (CTEB)** built in this paper. We manually re-annotate the sentences from validation set and test set of CaRB[3] (Bhardwaj et al., 2019) and a part of sentences of LSOIE[4] (Solawetz and Larson, 2021) dataset. BIO annotation is used to label subjects, relations, objects, and constraints in sentences from CTEB dataset, as described in Section 1 and Sec-

[3]https://github.com/dair-iitd/CaRB

[4]https://github.com/Jacobsolawetz/large-scale-oie

Table 2: Performance on CTEB and CaRB datasets

Models	CTEB				CaRB			
	F1	P	R	AUC	F1	P	R	AUC
OpenIE4 (Mausam, 2016)	57.0	66.1	50.1	34.1	48.8	-	-	27.2
BIO (Zhan and Zhao, 2020)	54.9	62.1	49.2	34.5	46.6	55.1	40.4	27.7
SpanOIE (Zhan and Zhao, 2020)	59.2	68.5	52.0	37.1	49.4	60.9	41.6	30.0
BERT+BiLSTM (Ro et al., 2020)	59.8	68.0	53.4	38.1	50.6	61.3	43.1	30.6
Multi^{2}OIE (Ro et al., 2020)	61.3	69.3	55.0	40.4	52.3	60.9	45.8	32.6
SRL_BERT (Solawetz and Larson, 2021)	60.4	68.2	54.2	39.3	50.9	59.6	44.4	31.3
OIE@OIA (Wang et al., 2022)	60.7	68.8	54.4	41.2	51.1	-	-	33.9
DetIE (Vasilkovsky et al., 2022)	61.3	69.8	54.6	42.7	52.1	-	-	**36.7**
Our IAN model	**63.8**	**71.9**	**57.4**	**44.5**	**54.6**	**63.5**	**47.9**	36.4

tion 3.1. A total of 2174 sentences with 3656 constrained knowledge tuples are annotated for the CTE task. We detail the annotation procedure in Appendix A.2. The dataset is divided into training set, validation set and test set using the split of 6:2:2. The second dataset is the commonly used CaRB dataset for Open IE task, which contains predicates and the corresponding arguments from 1282 sentences.

Transfer learning is utilized to train and evaluate the CTE task. The bootstrapped OpenIE4 dataset[5] is used as the training set in the first training stage. The model is first trained on a subset of OpenIE4 training set, which can improve the training speed. Then the model is trained and evaluated on CTEB dataset using transfer learning. Moreover, 1,748,826 constrained knowledge tuples are extracted from the 1,109,411 sentences in OpenIE4 dataset by utilizing the proposed IAN model, providing a training set for future works. The number N of adjacency matrices maintained for each matched span is set to 3. Additional experimental details are listed in Appendix A.1.

Evaluation metrics: P(Precision), R (Recall), F1 (F1-score), and AUC (the area under the curve) are used to evaluate the performance of different models. *Tuple match*[6] (Bhardwaj et al., 2019) is used as the matching function.

4.2 Experimental Results

The proposed IAN model is compared with the following models: **OpenIE4** (Mausam, 2016), **SpanOIE** (Zhan and Zhao, 2020), **BIO** (Zhan and Zhao, 2020), **Multi^{2}OIE** (Ro et al., 2020), **BERT+BiLSTM** (Ro et al., 2020), **SRL_BERT** (Solawetz and Larson, 2021), **OIE@OIA** (Wang et al., 2022) and **DetIE** (Vasilkovsky et al., 2022). OpenIE4 is a traditional rule-based Open IE method, and the other models extract tuples based on neural networks.

Table 2 shows the performances of different models on CTEB and CaRB datasets. Those two datasets are used to evaluate CTE task and Open IE task, respectively. From the experimental results in Table 2, we can observe that:

1) By exploiting the combinatorial interactions among context-specific external features and distinct-granularity internal features, the proposed IAN model outperforms other models on both CTE and Open IE tasks. In terms of F1, compared with the state-of-the-art methods DetIE and Multi^{2}OIE, our IAN achieves the best performance at 63.8% and 54.6% with increases of 2.5% and 2.3% on CTE and Open IE tasks. The superiorities of IAN mainly include: a) it introduces context-specific external features; b) distinct-granularity internal features are extracted from the raw text to sufficiently mine the inherent textual features; c) the combinatorial interactions between any two features in above external and internal features are effectively mined.

2) In both CTE and Open IE tasks, there is a gap between the metrics R (recall) and P (precision). It indicates that the number of tuples extracted from raw text is usually insufficient compared to the gold tuples.

3) Compared with the Multi^{2}OIE model, the main differences are that our IAN model leverages

[5]https://github.com/zhanjunlang/Span_OIE
[6]https://github.com/dair-iitd/CaRB

external and internal features and alleviates the interaction deficiency and the attention sparsity problems. The superiority of the IAN model shows the effectiveness of rich semantic features, combinatorial interactions and dominating attentions selection in information extraction tasks.

4.3 Ablation Study

To illustrate the contributions of different modules in the IAN model, we design ablation experiments on the IAN model for CTE task. The ablation study results are shown in Table 3. "w/o external" means removing the context-specific external features. "w/o interact" model uses general multi-head attention rather than the distribution-sparse multi-head attention. "w/o internal" model removes phrase-level, sentence-level, word-level and heterogeneous fused features. "w/o fuse" denotes removing the heterogeneous fused feature. "IAN-DBPedia" and "IAN-YAGO" represent using external knowledge graphs DBPedia and YAGO instead of Wikidata when generating external features.

Table 3: Ablation study for CTE task.

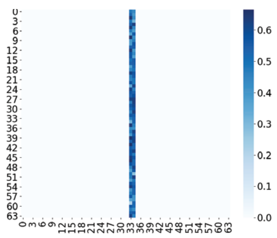
Models	CTEB			
	F1	P	R	AUC
w/o external	62.59	70.75	56.11	42.76
w/o internal	63.12	71.34	56.59	43.57
w/o fuse	63.18	71.43	56.64	43.69
w/o interact	61.96	70.08	55.53	42.12
IAN-DBPedia	63.48	71.53	57.06	44.18
IAN-YAGO	63.52	71.56	57.10	44.21
Our IAN	**63.83**	**71.85**	**57.41**	**44.52**

From Table 3, it can be seen that the performance of the IAN model degrades significantly when removing the distribution-sparse multi-head attention, indicating that combinatorial interactions and attention sparsity affects the performance of constrained tuple extraction. The "w/o external" model performs worse than the "w/o internal" model, which illustrates that the contribution of external features is larger than that of internal features. Additionally, the IAN model is not sensitive to the choices of different external knowledge bases.

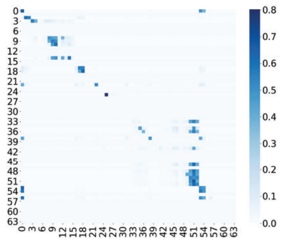
4.4 Visualization

The combinatorial interactions among the context-specific external features and the distinct-granularity internal features which utilize general multi-head self-attention and distribution-sparse multi-head attention in IAN model are visualized in Figure 2(a) and Figure 2(b), respectively.

It can be observed that interactions occur only among minority specific features in general multi-head attention, as shown in Figure 2(a). It is difficult for external and internal features to interact with each other. In contrast, for the distribution-sparse multi-head attention in Figure 2(b), interactions conduct among multiple features, and only the selected dominating query-key pairs are calculated. The distribution-sparse multi-head attention can effectively facilitate effective interactions and reduce the computation and memory usage.



(a) General Multi-head Attention



(b) Distribution-sparse Multi-Head Attention

Figure 2: Attention Visualization.

5 Conclusion and Future Work

In this paper, a novel task called Constrained Tuple Extraction (CTE) has been proposed, which aims to guarantee the validity of knowledge tuples and represent the constrained information in the real world more accurately. Interaction-aware network is designed to fulfill CTE task, which can effectively extract constrained knowledge tuples from raw texts. The proposed network can introduce the context-specific external features, sufficiently extract the distinct-granularity internal features from texts, and effectively exploit the combinatorial interactions among external and internal features. Meanwhile, distribution-sparse multi-head attention is developed to facilitate combinatorial interactions and alleviate the interaction deficiency and attention sparsity problems. Extensive experiments demonstrate that the proposed IAN model outperforms present methods. In the future, we will address the issues about the vertical domain-oriented constrained tuple extraction.

Limitations

The open information extraction methods may amplify the bias of the corpus by extracting any relation occurring in the data. The models with deep learning may learn the relation bias from the training corpus and extract those biased statements. To mitigate the effect of data bias, we try to balance the relations in constrained tuples and the ratio of constraints when constructing the CTEB dataset. In addition, the utilization of external auxiliary information increases additional computation time. Our IAN model has still achieved superior performance when the external auxiliary information is removed.

Acknowledgements

The work is supported by the National Key Research and Development Program of China (2020AAA0104903) and the National Natural Science Foundation of China (62072039). We thank the anonymous reviewers for their valuable comments and feedback.

References

Gabor Angeli, Melvin Jose Johnson Premkumar, and Christopher D. Manning. 2015. Leveraging linguistic structure for open domain information extraction. In *ACL*, pages 344–354.

Pratyay Banerjee and Chitta Baral. 2020. Self-supervised knowledge triplet learning for zero-shot question answering. In *EMNLP*, pages 151–162.

Sangnie Bhardwaj, Samarth Aggarwal, and Mausam. 2019. Carb: A crowdsourced benchmark for open IE. In *EMNLP/IJCNLP*, pages 6261–6266.

Giuseppe Carlo Calafiore, Stephane Gaubert, and Corrado Possieri. 2020. Log-sum-exp neural networks and posynomial models for convex and log-log-convex data. *IEEE Trans. Neural Networks Learn. Syst.*, 31(3):827–838.

Kai Chen, Ye Wang, Yitong Li, and Aiping Li. 2022. Rotateqvs: Representing temporal information as rotations in quaternion vector space for temporal knowledge graph completion. In *ACL*, pages 5843–5857.

Luciano Del Corro and Rainer Gemulla. 2013. Clausie: clause-based open information extraction. In *WWW*, pages 355–366.

Lei Cui, Furu Wei, and Ming Zhou. 2018. Neural open information extraction. In *ACL*, pages 407–413.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, pages 4171–4186.

Anthony Fader, Stephen Soderland, and Oren Etzioni. 2011. Identifying relations for open information extraction. In *EMNLP*, pages 1535–1545.

Xiaoyi Fu, Jie Zhang, Hao Yu, Jiachen Li, Dong Chen, Jie Yuan, and Xindong Wu. 2020. A speech-to-knowledge-graph construction system. In *IJCAI*, pages 5303–5305.

Tony Gracious, Shubham Gupta, Arun Kanthali, Rui M. Castro, and Ambedkar Dukkipati. 2021. Neural latent space model for dynamic networks and temporal knowledge graphs. In *AAAI*, pages 4054–4062.

Tianwen Jiang, Tong Zhao, Bing Qin, Ting Liu, Nitesh V. Chawla, and Meng Jiang. 2019. The role of: A novel scientific knowledge graph representation and construction model. In *KDD*, pages 1634–1642.

Zhengbao Jiang, Wei Xu, Jun Araki, and Graham Neubig. 2020. Generalizing natural language analysis through span-relation representations. In *ACL*, pages 2120–2133.

Jaehun Jung, Jinhong Jung, and U Kang. 2021. Learning to walk across time for interpretable temporal knowledge graph completion. In *KDD*, pages 786–795.

Keshav Kolluru, Vaibhav Adlakha, Samarth Aggarwal, Mausam, and Soumen Chakrabarti. 2020a. Openie6: Iterative grid labeling and coordination analysis for open information extraction. In *EMNLP*, pages 3748–3761.

Keshav Kolluru, Samarth Aggarwal, Vipul Rathore, Mausam, and Soumen Chakrabarti. 2020b. Imojie: Iterative memory-based joint open information extraction. In *ACL*, pages 5871–5886.

Feng-Lin Li, Hehong Chen, Guohai Xu, Tian Qiu, Feng Ji, Ji Zhang, and Haiqing Chen. 2020. Alimekg: Domain knowledge graph construction and application in e-commerce. In *CIKM*, pages 2581–2588.

Zixuan Li, Saiping Guan, Xiaolong Jin, Weihua Peng, Yajuan Lyu, Yong Zhu, Long Bai, Wei Li, Jiafeng Guo, and Xueqi Cheng. 2022. Complex evolutional pattern learning for temporal knowledge graph reasoning. In *ACL*, pages 290–296.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *ICLR (Poster)*.

Mausam. 2016. Open information extraction systems and downstream applications. In *IJCAI*, pages 4074–4077.

Deepak Nathani, Jatin Chauhan, Charu Sharma, and Manohar Kaul. 2019. Learning attention-based embeddings for relation prediction in knowledge graphs. In *ACL*, pages 4710–4723.

Youngbin Ro, Yukyung Lee, and Pilsung Kang. 2020. Multi^{2}oie: Multilingual open information extraction based on multi-head attention with BERT. In *EMNLP (Findings)*, volume EMNLP 2020 of *Findings of ACL*, pages 1107–1117.

Arpita Roy, Youngja Park, Taesung Lee, and Shimei Pan. 2019. Supervising unsupervised open information extraction models. In *EMNLP/IJCNLP*, pages 728–737.

Jingbo Shang, Jialu Liu, Meng Jiang, Xiang Ren, Clare R. Voss, and Jiawei Han. 2018. Automated phrase mining from massive text corpora. *IEEE Trans. Knowl. Data Eng.*, 30(10):1825–1837.

Jacob Solawetz and Stefan Larson. 2021. LSOIE: A large-scale dataset for supervised open information extraction. In *EACL*, pages 2595–2600.

Gabriel Stanovsky, Julian Michael, Luke Zettlemoyer, and Ido Dagan. Supervised open information extraction. In *NAACL-HLT*, pages 885–895.

Gabriel Stanovsky, Julian Michael, Luke Zettlemoyer, and Ido Dagan. 2018. Supervised open information extraction. In *NAACL-HLT*, pages 885–895.

Mingming Sun, Xu Li, Xin Wang, Miao Fan, Yue Feng, and Ping Li. 2018. Logician: A unified end-to-end neural approach for open-domain information extraction. In *WSDM*, pages 556–564.

Yao-Hung Hubert Tsai, Shaojie Bai, Makoto Yamada, Louis-Philippe Morency, and Ruslan Salakhutdinov. 2019. Transformer dissection: An unified understanding for transformer's attention via the lens of kernel. In *EMNLP/IJCNLP*, pages 4343–4352.

Michael Vasilkovsky, Anton Alekseev, Valentin Malykh, Ilya Shenbin, Elena Tutubalina, Dmitriy Salikhov, Mikhail Stepnov, Andrey Chertok, and Sergey I. Nikolenko. 2022. Detie: Multilingual open information extraction inspired by object detection. In *AAAI*, pages 11412–11420.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*, pages 5998–6008.

Xin Wang, Minlong Peng, Mingming Sun, and Ping Li. 2022. Oieoia: an adaptable and efficient open information extraction framework. In *ACL*, pages 6213–6226.

Fuzhao Xue, Aixin Sun, Hao Zhang, and Eng Siong Chng. 2021. Gdpnet: Refining latent multi-view graph for relation extraction. In *AAAI*, pages 14194–14202.

Xiaojun Xue, Chunxia Zhang, Zhendong Niu, and Xindong Wu. 2023. Multi-level attention map network for multimodal sentiment analysis. *IEEE Trans. Knowl. Data Eng.*, 35(5):5105–5118.

Mohamed Yahya, Steven Whang, Rahul Gupta, and Alon Y. Halevy. 2014. Renoun: Fact extraction for nominal attributes. In *EMNLP*, pages 325–335.

Zhou Yu, Jun Yu, Chenchao Xiang, Jianping Fan, and Dacheng Tao. 2018. Beyond bilinear: Generalized multimodal factorized high-order pooling for visual question answering. *IEEE Trans. Neural Networks Learn. Syst.*, 29(12):5947–5959.

Junlang Zhan and Hai Zhao. 2020. Span model for open information extraction on accurate corpus. In *AAAI*, pages 9523–9530.

Zexuan Zhong and Danqi Chen. 2021. A frustratingly easy approach for entity and relation extraction. In *NAACL-HLT*, pages 50–61.

Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. 2021. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *AAAI*, pages 11106–11115.

A Appendix

A.1 Experimental Details

Datasates and Training. For the CTEB dataset, we reproduce the baselines. For the CaRB dataset, we report the experimental results from the papers of the baselines. To enhance the training process, transfer learning is used to train the proposed interaction-aware network (IAN). First, a subset containing 20,000 sentences from OpenIE4 dataset is used to train the IAN model. Then, the output layer of the IAN model is modified to the form required by the constrained tuple extraction (CTE) task. Next, the IAN model is trained, validated and tested on the newly annotated Constraint Tuples Extraction Benchmark (CTEB) dataset. For the CaRB dataset, all the models (including our IAN model) are trained the same way as the existing Open IE works, and are not trained on the CTEB dataset. For the CTEB dataset, all the deep learning models (including our IAN model) are trained on the OpenIE4 training set and the training set of the CTEB dataset, and then the results are reported on the CTEB test set. We ensure that our model and the baseline models are trained and evaluated in the same way to guarantee fair comparisons.

The statistics of the datasets used in this paper are shown in Table 4.

Parameter Settings. AdamW optimizer (Loshchilov and Hutter, 2019) is used to train the IAN model. The initial learning rate is set to 3e-5. The steps of warmup is set to one-tenth of the total training steps. The epochs for the

Table 4: Sentences and tuples statistics in each dataset. † and ‡ denote bootstrap and manual annotated datasets, respectively.

Dataset	# Sents.	# Tuples
OpenIE4†	1,109,411	2,175,294
CTEB‡	2174	3656
CaRB-dev‡	641	2548
CaRB-test‡	641	2715

first and second training stages are set to 20 and 100, respectively. The IAN model includes 4 distribution-sparse multi-head attention blocks with 8 attention heads. The dropout rates for the attention blocks and classification layer are set to 0.05 and 0.2, respectively. The constant sampling factor c in distribution-sparse multi-head attention is set to 5. Pytorch is used to implement the IAN model. We have conducted all experiments on the Ubuntu system with 3090 GPU.

Data Bias Mitigation. To mitigate the effect of data bias, we try to balance the relations in constrained tuples and the ratio of constraints when constructing the CTEB dataset. The ratio of temporal, spatial, and conditional constrained tuples is 845:465:246. During evaluation, the CTEB dataset is randomly divided into training, validation, and test set to alleviate the data bias. During training, we first utilize transfer learning to perform warm up training on the large Open IE dataset OpenIE4, and then train the model on the CTEB training set, which can also alleviate the bias problem.

Other Details. As for the implementation details of the competing models, the inputs have not been changed, and only the output layer is modified. The output layer is changed from the original classification of predicate and arguments to the classification of subject, relation, object, and three constraints. The rest of the model details have not been changed. Experimental results reported in this paper are averaged over three different seed settings. For the ablation experiments, the hyper-parameter settings are the same as those of the final model, except for the removal of specific modules.

In this work, AUC (Area Under Curve) is calculated from a plot of the precision and recall values for all potential cutoffs. Matching a tuple accurately is challenging. The reason lies in that both constrained tuple extraction and Open IE tasks require how to match different spans containing multiple words in a sentence.

From the experimental results, we can observe that the low AUCs is due to the low recall values. For the proposed IAN model, the precision achieves 0.719 and 0.635 on the CTEB dataset and the CaRB dataset respectively, while the recalls are 0.574 and 0.479. This fact indicates that our model predicts most correct tuples in the predicted results, however, it does not cover the ground truth tuples well. Th reason lies in that there are multiple true tuples in each sample, and each tuple contains spans of different elements including subjects, relations, objects and constraints. Therefore, there are many complex elements to be predicted in each sentence. However, our IAN model has outperformed the state-of-the-art methods.

In the model design, the constraint identification is taken as a process of multi-class classification. Architecturally, in the head of our designed IAN model, there is a layer to predict the category of the constraints. That is, the three kinds of constraints are regarded as different classes, and we employed a fully connected layer in our IAN model to achieve this goal. Therefore, our proposed model is applicable to other constraints, and only needs to modify the classes of the constraints in the output layer. In this way, our IAN model renders extensibility in usage.

A.2 Annotation Procedure

The sentences in the CTEB dataset are selected from the CaRB dataset and the LSOIE dataset. When selecting sentences, we try to pick sentences that contain constraints as much as possible, and do not make any changes to the sentences for subsequent annotation. Each sample was annotated by three graduate students, one of whom did the preliminary annotation, and the other two checked and corrected the annotations.

We use brat annotation tool[7] to annotate as many constraints as possible in the selected sentences. Three steps are performed when performing annotation task: 1) identifying subjects, relations, and objects; 2) identifying constraints, including temporal, spatial, and conditional constraints; 3) combining subjects, relations, objects, and constraints to acquire the constraint tuples.

We abide by the following principles when annotating sentences: 1) completeness: all subjects,

[7] http://brat.nlplab.org/

relations, objects and constraints in the sentences need to be annotated; 2) assertedness: all constraint tuples are implied by the original sentences; 3) atomicity: the quadruples are used as the atomic tuples.

When there are multiple constraints or multiple (subject, relation, object) triples or both in a sentence, one related constraint is added to each (subject, relation, object) triple. When (subject, relation, object) triples and constraints exist in a many-to-many situation, the copy operation for triples and constraints is performed so that the quadruples are still atomic tuples.

A.3 External Feature

To utilize external features, additional external knowledge related to the input text needs to be acquired. The purpose of choosing Wikidata is that Wikidata can provide the external knowledge related to the spans in the sentence. The external knowledge provided by Wikidata includes the properties and entities related to the spans.

To obtain external auxiliary information, we enumerate the spans in the sentence setting a maximum length (5 words in this work), and then call the query interface of Wikidata by inputting the enumerated spans. Next, the spans that can receive a response containing contents from the interface are regarded as the aligned spans. In other words, aligned spans refer to the spans that can be queried as nodes in Wikidata.

In theory, the complexity of maintaining N adjacency matrices is $O(n^3)$. In fact, since the amount of the nodes in external auxiliary information is not large, the calculation consumption is acceptable. The average number of matched spans in each sample is about 1.5. The number of nodes contained in the external information for each matched span is 20 (5 nodes for the first hop, 15 nodes for the second hop). N represents the number of adjacency matrices maintained for each matched span, and is set to 3 in the experiments. Thus, the calculation of maintaining N adjacency matrices for each sample is about $1.5 * (3 + 1) * (20 + 1) * (20 + 1)$.

"N multi-views" represents N potential external knowledge graph structures. Previous related works built a fixed external knowledge graph structure for input text. Comparatively, we generate N distinguishable external knowledge graph structures for spans in text via N multi-views. Typically, each view selects the contributed edges and nodes from the original two-hop external knowledge graph structure, which is accomplished by utilizing sentence contextual information to generate Gaussian distributions and weighting the edges based on the KL divergence. Thereby, "N multi-views" can generate non-fixed context-specific external knowledge graph structures for spans.

In this way, external auxiliary information can be used more effectively and flexibly by selecting sentence context-appropriate external knowledge. In addition, too many multi-views could induce redundant information, and the generation of N multi-views requires certain computing resources. After balancing model performance and computation consumption, the number of generated multi-views is set to 3.

The embedding generation method KBGAT (Nathani et al., 2019) can generate embeddings for nodes and edges in a graph, which considers both the weights of nodes and edges. The context-specific external feature F_k is generated as a node in graph G_m. The entire construction step of the external feature F_k is done independent of the training and test sets given.

Table 5 shows the example case about external information usage. Without external information, the model extracts "system" as the subject. By utilizing external information, the model can know that "DDB" is short for "Distributed Database" and identifies "DDB system" as the subject.

Table 5: Example case about external information usage.

Sentences	Tuples
In general, DDB systems use smaller computer systems.	*without external features:(systems, use, smaller computer systems, in general)*
	with external features:(DDB systems, use, smaller computer systems, in general)

A.4 Ralated Work

The main distinctive work in our paper is to concatenate the constraints into the traditional tuples to enhance the validity of the extracted knowledge tuples. Technically, as constraints are usually written into the sentences with various latent forms, extracting them correctly from free texts with no semantic conflicts is a challenging task. To this end, we propose the constrained tuple extraction task.

As for the Open IE, it mainly concerns about how to extract the predicates and the corresponding arguments. By contrast, in our work, we focus on how to extract the constrained tuples, which are formatted as (subject, relation, object, constraint). In other words, we emphasize the validity of knowledge tuples by introducing constrains, which provides a standardized and unified representation for knowledge tuples.

We adopt the two-stage extraction strategy similar to the Multi2OIE model. Our IAN model facilitates the combinatorial interactions among the context-specific external features and the distinct-granularity internal features to effectively mine the potential constraints in knowledge.

The difference between our IAN model and the Multi2OIE model is that the Multi2OIE model simply exploits BERT representations and predicate features to extract tuples, while our IAN model mines the combinatorial interactions among the external and internal features.

More specifically, Multi2OIE model feeds predicate feature, positional embedding, and BERT representation into multi-head attention blocks for Open IE tuple extraction. In our proposed IAN model, we design distribution-sparse multi-head attention to select the dominating attentions, and feed distinct-granularity internal features and context-specific external features to distribution-sparse multi-head attention to mine the combinatorial interactions for constrained tuple extraction. Concretely, distinct-granularity internal features include word-level, sentence-level, phrase-level textual features, BERT hidden, heterogeneous fused feature, predicate feature and position embedding.

The constrained tuple extraction task is actually a fundamental task for constructing the constrained knowledge graph construction with good quality. Technically, the phrases in sentences that act as subjects and objects in constrained tuples are usually with the type of entities. Thereby, in this situation, the form of the constrained tuples is " (head entity, relation, tail entity, constraint) ". Accordingly, these constraint tuples (i.e., quadruples) can be transformed into triples and their constraints. Hence, on the one hand, triples are the basic components of knowledge graphs. On the other hand, constrained tuples with different kinds of constrains constitute a specific kind of constrained knowledge graph (for example, temporal knowledge graph) (Chen et al., 2022; Gracious et al., 2021).

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
Section "Limitations"

☐ A2. Did you discuss any potential risks of your work?
Not applicable. Left blank.

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

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

B ☑ Did you use or create scientific artifacts?

Section 1,2,3,4,Appendix

☑ B1. Did you cite the creators of artifacts you used?
Section 1,2,3,4

☑ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
Section 1,3

☑ 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)?
Section 3,4

☐ 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. Section 4, Appendix

☑ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Section 4

☑ 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.
Section 4

C ☑ Did you run computational experiments?

Left blank.

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Section 4, Appendix

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?
Section 4, Appendix

☑ 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?
Section 4

☑ 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.)?
Appendix

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

Section 4, Appendix

☐ 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. Appendix

☐ 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.