A Canonicalization-Enhanced Known Fact-Aware Framework For Open Knowledge Graph Link Prediction

Yilin Wang[1] , **Minghao Hu**[2*] , **Zhen Huang**[1*] , **Dongsheng Li**[1] , **Wei Luo**[2] , **Dong Yang**[1] and **Xicheng Lu**[1]
[1]National University of Defense Technology
[2]Information Research Center of Military Science
{wangyilin14, huangzhen, dsli, yangdong14, xclu}@nudt.edu.cn, huminghao16@gmail.com, htqxjj@126.com

Abstract

Open knowledge graph (OpenKG) link prediction aims to predict missing factual triples in the form of *(head noun phrase, relation phrase, tail noun phrase)*. Since triples are not canonicalized, previous methods either focus on canonicalizing noun phrases (NPs) to reduce graph sparsity, or utilize textual forms to improve type compatibility. However, they neglect to canonicalize relation phrases (RPs) and triples, making OpenKG maintain high sparsity and impeding the performance. To address the above issues, we propose a Canonicalization-Enhanced Known Fact-Aware (**CEKFA**) framework that boosts link prediction performance through sparsity reduction of RPs and triples. First, we propose a similarity-driven RP canonicalization method to reduce RPs' sparsity by sharing knowledge of semantically similar ones. Second, to reduce the sparsity of triples, a known fact-aware triple canonicalization method is designed to retrieve relevant known facts from training data. Finally, these two types of canonical information are integrated into a general two-stage re-ranking framework that can be applied to most existing knowledge graph embedding methods. Experiment results on two OpenKG datasets, ReVerb20K and ReVerb45K, show that our approach achieves state-of-the-art results. Extensive experimental analyses illustrate the effectiveness and generalization ability of the proposed framework.

1 Introduction

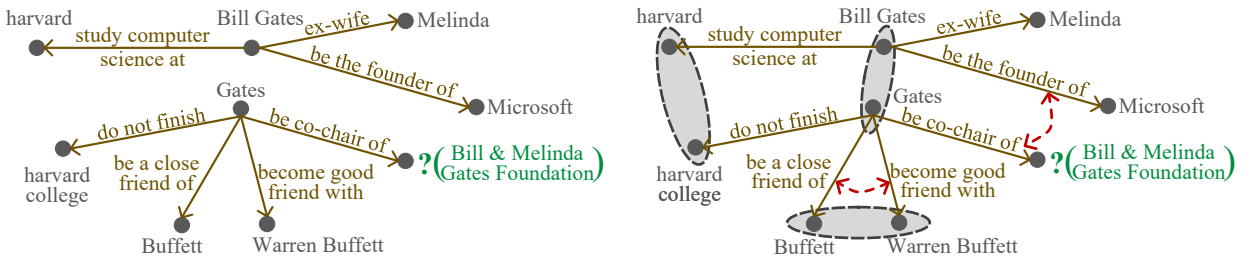
Open knowledge graph (OpenKG) stores factual triples automatically extracted from text corpus by open information extraction (OpenIE) approaches, such as ReVerb [Fader *et al.*, 2011] and CALMIE [Saha and others, 2018]. These triples are in the form of *(head noun phrase, relation phrase, tail noun phrase)*, e.g., *(Bill Gates, be the founder of, Microsoft)*. OpenKG is highly adaptable and can be easily exploited by new domain-specific corpora since no pre-specified ontologies are required. However, OpenKG may not be suitable for directly used by an end task owing to their sparsity and incompleteness. Therefore, tasks such as OpenKG canonicalization [Lin and Chen, 2019; Dash *et al.*, 2021] and OpenKG linking [Dubey *et al.*, 2018; Liu *et al.*, 2021] are studied to reduce graph sparsity, while OpenKG link prediction task [Gupta *et al.*, 2019; Chandrahas and Talukdar, 2021] is proposed to reduce graph incompleteness. Since OpenKG linking may be unavailable when bootstrapping OpenKG from a domain-specific corpus that lacks pre-defined entities and relations, we focus on utilizing OpenKG canonicalization for graph sparsity reduction to improve the performance of OpenKG link prediction.

Unlike curated knowledge graph (CKG), e.g., Freebase [Bollacker *et al.*, 2008], OpenKG is extremely sparse since noun phrases (NPs) and relation phrases (NPs) are not canonicalized. As a result, severe long-tail problems exist in embedding-based link prediction methods, such as TransE [Bordes *et al.*, 2013] and ConvE [Dettmers *et al.*, 2018], where embeddings of infrequent phrases are poorly learned due to insufficient training. Recently, OpenKG link prediction starts to receive more and more attention owing to its efficient learning ability in high-sparsity situations. CaRe [Gupta *et al.*, 2019] utilizes NPs' canonical clusters to share NP embeddings and encodes RPs' semantic meanings by GRU [Cho *et al.*, 2014]. OKGIT [Chandrahas and Talukdar, 2021] employs a pre-trained BERT [Devlin *et al.*, 2019] to provide type guidance for NPs. OKGSE [Xie *et al.*, 2022] learns separate parts of embedding for NP and its cluster, and captures knowledge among NPs' neighboring triples. Nevertheless, there are still two limitations that hinder the performance of OpenKG link prediction.

First, previous methods have not taken into account RP canonicalization explicitly. The number of RPs can reach tens of thousands because of their diverse expressions, yet many of them may refer to the same semantic, such as "be a close friend of" and "become good friend with" shown in Fig. 1. In order to correlate these RPs, CaRe [Gupta *et al.*, 2019] encodes tokens of RPs initialized with glove vectors [Pennington *et al.*, 2014]. However, there is clearly a semantic gap between glove embeddings and randomly initialized NP embeddings, and it neglects to canonicalize RPs explicitly by sharing their semantics.

Second, triple canonicalization, which directly groups semantically similar triples, has been ignored by existing

*Corresponding author



(a) OpenKG without canonical information

(b) OpenKG with various canonical information

Figure 1: Comparison of OpenKG w/o and w/ canonical information, where nodes are NPs, edges are RPs, and the green marker is the tail NP in missing fact *(Gates, be co-chair of, ?)*. (a) OpenKG is sparse, where similar NPs/RPs are stored separately due to the lack of canonicalization. (b) The sparsity of OpenKG can be reduced by taking into account the canonicalization of NPs (dashed ellipses), RPs (red dashed arrows), and their composed triples, so that the knowledge of related but distinct NPs, RPs, and triples can be shared.

works. Prior approaches [Kolluru *et al.*, 2021; Lovelace *et al.*, 2021] use textual information from triples that consist of query and promising candidate tails for re-ranking in CKG. However, directly applying these methods to OpenKG fails to utilize knowledge from triples with similar semantics, where relevant known triples can provide helpful contextual information for predicting the missing tail. Taking Fig. 1(b) as an example, it will be easier to infer the tail NP of *(Gates, be co-chair of, ?)* is "Bill & Melinda Gates Foundation" by sharing known triple knowledge from *(Bill Gates, be the founder of, Microsoft)* and *(Bill Gates, ex-wife, Melinda)*.

To tackle these issues, we propose a **C**anonicalization-**E**nhanced **K**nown **F**act-**A**ware (**CEKFA**) framework, which improves OpenKG link prediction through sparsity reduction of RPs and triples. First, we propose a similarity-driven RP canonicalization method to collect semantically similar RPs for each RP as its canonical information. Second, we introduce a known fact-aware triple canonicalization method through retrieving known facts associated with the query from training data as its canonical triples. Finally, a general two-stage framework consisting of a canonicalization-enhanced encoder and a known fact-aware re-ranker is proposed to utilize these two kinds of canonical information for link prediction. The advantages of our approach are that the proposed canonicalization methods require neither predefined knowledge bases for OpenKG linking nor several external tools for generating various side information [Vashishth *et al.*, 2018; Liu *et al.*, 2021], and can be easily adapted to different query encoding models such as TransE [Bordes *et al.*, 2013], ComplEx [Trouillon *et al.*, 2016], ConvE [Dettmers *et al.*, 2018] and so on. In summary, Our main contributions are:

- We propose a two-stage re-ranking framework named CEKFA to boost the performance of OpenKG link prediction through reducing the sparsity of RPs and triples.
- We develop novel and effective similarity-driven RP canonicalization and known fact-aware triple canonicalization methods for sparsity reduction.
- We achieve state-of-the-art results on two OpenKG datasets, ReVerb20K and ReVerb45K, and obtain different degrees of improvement when CEKFA is applied to five typical query encoding models. Ablation studies and extensive analyses demonstrate the effectiveness and scalability of our framework. Source codes and datasets of this paper are available at https://github.com/wylResearch/CEKFA.

2 Related Work

2.1 OpenKG Link Prediction

Knowledge graph embedding (KGE) methods have been commonly explored for the link prediction problem in CKG, typically including distance-based models [Bordes *et al.*, 2013; Lin *et al.*, 2015; Sun *et al.*, 2019], semantic matching models [Trouillon *et al.*, 2016; Zhang *et al.*, 2019], and neural network-based models [Dettmers *et al.*, 2018; Schlichtkrull *et al.*, 2018].

However, link prediction for OpenKG is a relatively under-explored area. CESI [Vashishth *et al.*, 2018] focuses on obtaining multiple side information of NPs and RPs for embedding learning built on HolE [Nickel *et al.*, 2016], so as to perform canonicalization task instead of link prediction. Based on ConvE [Dettmers *et al.*, 2018], CaRe [Gupta *et al.*, 2019] takes the canonicalization of NPs into account to enhance NP embeddings, and learns RP embeddings by GRU [Cho *et al.*, 2014]. Instead of concentrating on canonicalization, OKGIT [Chandrahas and Talukdar, 2021] employs pre-trained BERT [Devlin *et al.*, 2019] to improve the type compatibility of tail NPs. [Broscheit *et al.*, 2020] provides an OpenKG benchmark called OLPBENCH, and employs ComplEx [Trouillon *et al.*, 2016] to learn embeddings initialized by LSTM [Hochreiter and Schmidhuber, 1997]. OKGSE[Xie *et al.*, 2022] learns cluster-segregated NP embeddings enhanced by neighboring contexts and type compatibility. Unlike the above methods, we focus on reducing the sparsity of RPs and triples through obtaining their canonical information to improve link prediction performance, which requires neither predefined relations for OpenKG linking nor additional side information for OpenKG canonicalization [Vashishth *et al.*, 2018; Liu *et al.*, 2021].

2.2 Re-ranking for Link Prediction

By exploiting textual information, re-ranking promising candidates through PLMs brings performance gains for link pre-

diction in CKG [Kolluru *et al.*, 2021; Lovelace *et al.*, 2021]. CEAR [Kolluru *et al.*, 2021] applies BERT [Devlin *et al.*, 2019] as a re-ranker to score promising candidates together by concatenating textual forms of the query and all candidate tail entities into one input sentence. [Lovelace *et al.*, 2021] develops a student network based on BERT [Devlin *et al.*, 2019] with the textual concatenation of query and a candidate tail as input, which is distilled from their ranking model. Different from them, we not only utilize query along with tail candidates but also retrieve useful known facts related to the query from training set to assist in re-scoring, which is able to reduce the triple sparsity by sharing knowledge of similar triples and provide contextualized information.

2.3 Retrieving from Training Data

Lately, several methods show that explicitly retrieving from training data is useful for several NLP tasks [Gu *et al.*, 2018; Khandelwal *et al.*, 2020; Wang *et al.*, 2022]. [Wang *et al.*, 2022] develops a model pipeline that retrieves knowledge from training data for tasks such as summarization, machine translation, language modeling, and question answering, where the model input is concatenated with the retrieved content obtained by BM25 [Robertson and Zaragoza, 2009]. To the best of our knowledge, we are the first to retrieve training data for scoring promising candidates at the re-ranking stage for OpenKG link prediction.

3 CEKFA: The Proposed Framework

To solve the problems of failing to utilize the canonical information of RPs and triples for OpenKG link prediction, we propose the Canonicalization-Enhanced Known Fact-Aware (CEKFA) framework, shown in Fig. 2. First, we introduce two different canonicalization methods for sparsity reduction of RPs and triples in Sec. 3.2. To exploit the obtained canonical information, we design a *canonicalization-enhanced encoder* that utilizes canonicalization of both NPs and RPs to learn dense embeddings for scoring a triple in Sec. 3.3, and a *known fact-aware re-ranker* that adopts triple canonicalization to re-score promising candidates in Sec. 3.4.

3.1 Problem Formulation

Let $\mathcal{G} = (\mathcal{E}, \mathcal{R}, \mathcal{F})$ be an OpenKG, where $\mathcal{E}$ is the set of noun phrases (NPs), $\mathcal{R}$ is the set of relation phrases (RPs), and $\mathcal{F} = \{(e_h, r, e_t) | e_h \in \mathcal{E}, r \in \mathcal{R}, e_t \in \mathcal{E}\}$ is the set of facts. Here, e_h and e_t are the head NP and tail NP in a factual triple. Each NP $e \in \mathcal{E}$ and RP $r \in \mathcal{R}$ is a sequence of tokens, denoted as $x^e = (x_1^e, x_2^e, \ldots, x_{l_e}^e)$ and $x^r = (x_1^r, x_2^r, \ldots, x_{l_r}^r)$, respectively, where l_e and l_r are the numbers of tokens in NP e and RP r. Given a query $(e_h, r, ?)$, the link prediction task aims at inferring the tail NP e_t in missing fact (e_h, r, e_t) by learning the triple score representing the holding probability.

3.2 Sparsity Reduction for RP and Triple

To reduce the sparsity of RPs and triples for OpenKG link prediction, two canonicalization methods, namely *similarity-driven RP canonicalization* and *known fact-aware triple canonicalization*, are proposed.

Similarity-driven RP Canonicalization

Prior methods like CaRe [Gupta *et al.*, 2019] fail to explicitly canonicalize RPs, resulting in knowledge of similar RPs cannot be shared. Considering this, we propose to retrieve similar RPs for each RP as its canonical information, which does not require an existing knowledge base for OpenKG linking.

First, to obtain the semantic representation of RPs, each of them is constructed into an input sequence. Take RP r_j as an example, its input sequence s^{r_j} is set as "[ENT1] x^{r_j} [ENT2]", where "[ENT1]" and "[ENT2]" are two introduced tokens. Then, a pre-trained language model SBERT [Reimers and Gurevych, 2019] is employed to encode the input and obtain its dense sentence representation, denoted as function f_{SBERT}. Finally, to get the canonical RPs for r_j, a retrieval system $\mathcal{S}_{rp}$ is adopted to match its representation against all other RPs in $\mathcal{R}$ and return the Top-M_{rp} most similar ones:

$$\mathcal{S}_{rp}(r_j, \mathcal{R}) = \{r_1, \ldots, r_{M_{rp}}\} = N_r(r_j) \tag{1}$$

where M_{rp} is a hyperparameter and $N_r(r_j)$ is the canonical neighbor RPs of r_j. The semantic similarity $sim_{j,u}^{rp}$ between r_j and $r_u \in \mathcal{R}$ is computed as the cosine similarity between their sentence representations:

$$sim_{j,u}^{rp} = \cos(f_{SBERT}(s^{r_j}), f_{SBERT}(s^{r_u})) \tag{2}$$

where s^{r_u} is the input sequence of r_u.

Known Fact-aware Triple Canonicalization

Prior works ignore the sparsity reduction of triples in OpenKG, and they fail to utilize relevant known triples for providing useful hints. Triple canonicalization aims to retrieve these relevant triples with respect to the query, so as to deliver contextualized information for guiding the prediction of unseen queries. Motivated by [Wang *et al.*, 2022], we propose to canonicalize a query by retrieving similarly-expressed known facts from training data.

First, we index all known factual triples in the training set to a dictionary, i.e., $\mathcal{C} = \{(k, v)\}$, where the key k is the token concatenation of head NP and RP, and the value v is the token concatenation of head NP, RP, and tail NP. For instance, the k and v for a triple (e_h, r, e_t) are expressed as:

$$k = [x^{e_h}, x^r],\ v = [x^{e_h}, x^r, x^{e_t}] \tag{3}$$

where [] means concatenation, and x^{e_h}, x^r, x^{e_t} are tokens of e_h, r and e_t, respectively.

For a query $(e_h, r, ?)$, we identify known facts with similar query as its canonical triples, so as to provide clues for predicting the missing tail e_t. We match the key of the query, written as $k_q = [x^{e_h}, x^r]$, against keys of all known facts, and return the Top-M_{triple} most similar ones along with their values through a retrieval system $\mathcal{S}_{triple}$:

$$\mathcal{S}_{triple}(k_q, \mathcal{C}) = \{(k_1, v_1), \ldots, (k_{M_{triple}}, v_{M_{triple}})\} \tag{4}$$

where M_{triple} is a hyperparameter. $\mathcal{S}_{triple}$ is built based on SBERT [Reimers and Gurevych, 2019] as $\mathcal{S}_{rp}$ (Eq. (1)), in which the similarity $sim_{i,j}^{triple}$ between any two keys k_i and k_j can be formulated as follows:

$$sim_{i,j}^{triple} = \cos(f_{SBERT}(k_i), f_{SBERT}(k_j)) \tag{5}$$

At last, the canonical triples $N_{triple}(e_h, r, ?)$ for $(e_h, r, ?)$ are set to the values in the retrieved results $\mathcal{S}_{triple}(k_q, \mathcal{C})$:

$$N_{triple}(e_h, r, ?) = \{v_1, \ldots, v_{M_{triple}}\} \tag{6}$$

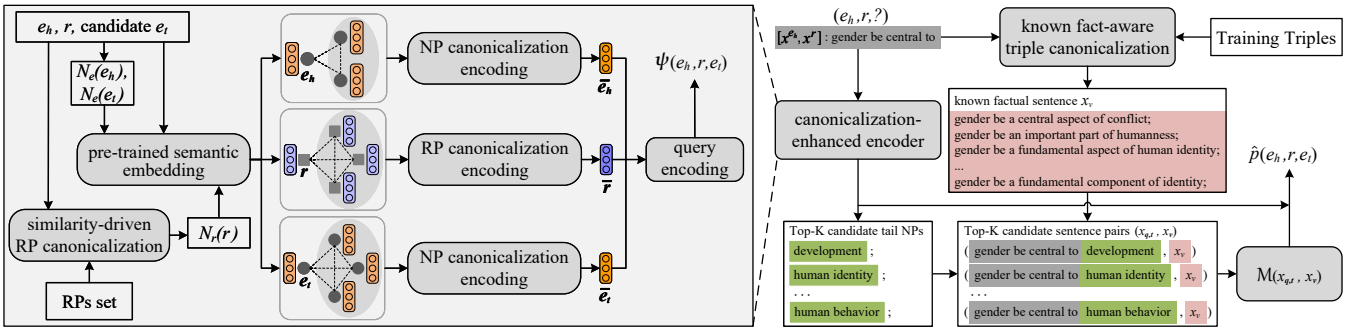


Figure 2: Overview of the proposed CEKFA framework. The left part shows the *canonicalization-enhanced encoder* for scoring a query triple (e_h, r, e_t) composed of $(e_h, r, ?)$ and a candidate tail NP e_t, which reduces graph sparsity by taking advantage of canonicalization for NPs and RPs (shown in shaded ellipse). The right part displays the process of the *known fact-aware re-ranker* for re-scoring predicted Top-K tail NPs, which utilizes triple canonicalization to reduce the sparsity of triples for providing contextualized hints.

3.3 Canonicalization-Enhanced Encoder

The *canonicalization-enhanced encoder* aims to sufficiently exploit the canonical information of NPs and RPs to score a triple (e_h, r, e_t), which consists of four modules: (1) *pre-trained semantic embedding*, of which embeddings are initialized with rich semantics rather than at random; (2) *NP canonicalization encoding*, which updates NP embeddings utilizing NPs' canonical information; (3) *RP canonicalization encoding*, which applies RPs' canonical information to update RP embeddings; (4) *query encoding*, which captures the interaction of a query triple and obtains its score.

Pre-trained Semantic Embedding

We employ pre-trained language models to encode the textual forms of NPs and RPs as initialization, which can identify similar NPs/RPs and give meaningful representations for long-tailed ones. For each NP/RP, its textual sequence is sent to the pre-trained BERT [Devlin *et al.*, 2019], and the output hidden states at the last layer are mean pooled to form the initialized embedding, denoted as function f_{BERT}. Then, initializing NP $e_i \in \mathcal{E}$ and RP $r_j \in \mathcal{R}$ to get their embeddings $\boldsymbol{e}_i \in \mathbb{R}^d$ and $\boldsymbol{r}_j \in \mathbb{R}^d$ can be written as $\boldsymbol{e}_i = f_{BERT}(x^{e_i})$ and $\boldsymbol{r}_j = f_{BERT}(x^{r_j})$, where d is the embedding dimension, x^{e_i} and x^{r_j} are token sequences of e_i and r_j, respectively.

NP Canonicalization Encoding

In order to densify the OpenKG by leveraging NP canonicalization, we follow CaRe [Gupta *et al.*, 2019] to employ a Local Averaging Network, which is a non-parametric message passing network, for updating NP embeddings. For each NP e_i, let $N_e(e_i)$ be its canonical neighbor NPs, namely NPs belonging to the same entity as e_i, which are first aggregated to obtain its canonical neighbor vector $\boldsymbol{e}_i^n \in \mathbb{R}^d$ as follows:

$$\boldsymbol{e}_i^n = \frac{1}{\mid N_e(e_i) \mid} \sum_{e_u \in N_e(e_i)} \boldsymbol{e}_u \tag{7}$$

Then, $\bar{\boldsymbol{e}}_i \in \mathbb{R}^d$ is obtained by averaging $\boldsymbol{e}_i$ and the canonical neighbor vector $\boldsymbol{e}_i^n$, i.e., $\bar{\boldsymbol{e}}_i = (\boldsymbol{e}_i + \boldsymbol{e}_i^n)/2$, which entails information about NPs linked to an entity.

RP Canonicalization Encoding

Taking one step further, we take into account the RP canonicalization $N_r(r_j)$ obtained in Eq. (1). The embedding of r_j is updated in the same way as NPs, where its canonical neighbor vector $\boldsymbol{r}_j^n \in \mathbb{R}^d$ of RP r_j is first calculated from $N_r(r_j)$:

$$\boldsymbol{r}_j^n = \frac{1}{\mid N_r(r_j) \mid} \sum_{r_u \in N_r(r_j)} \boldsymbol{r}_u \tag{8}$$

Then $\bar{\boldsymbol{r}}_j \in \mathbb{R}^d$ is averaged from $\boldsymbol{r}_j$ and $\boldsymbol{r}_j^n$, i.e., $\bar{\boldsymbol{r}}_j = (\boldsymbol{r}_j + \boldsymbol{r}_j^n)/2$, so that it carries the knowledge of RPs similar to r_j.

Query Encoding

Query encoding intends to encode a query triple (e_h, r, e_t) to obtain its triple score $\psi(e_h, r, e_t)$, notated as the score function SF. We utilize the updated embeddings that are enhanced by canonicalization, i.e. $\bar{\boldsymbol{e}}_h$, $\bar{\boldsymbol{r}}$, $\bar{\boldsymbol{e}}_t$, as inputs to SF:

$$\psi(e_h, r, e_t) = SF(\bar{\boldsymbol{e}}_h, \bar{\boldsymbol{r}}, \bar{\boldsymbol{e}}_t) \tag{9}$$

Note that most KGE models could be leveraged to implement SF, such as distance-based models like TransE [Bordes *et al.*, 2013], semantic matching models like ComplEx [Trouillon *et al.*, 2016], and neural network-based models like ConvE [Dettmers *et al.*, 2018]. For instance, the score function based on ConvE [Dettmers *et al.*, 2018] is:

$$SF(\bar{\boldsymbol{e}}_h, \bar{\boldsymbol{r}}, \bar{\boldsymbol{e}}_t)_{ConvE} = f(vec(f(g(\bar{\boldsymbol{e}}_h, \bar{\boldsymbol{r}}) * \omega))W)\bar{\boldsymbol{e}}_t \tag{10}$$

where g is a operation of reshaping embeddings into a 2D matrix, ω is convolutional filters, f is a non-linear function, vec is a operation of reshaping a tensor into a vector, and W is a linear transformation matrix.

Model Training and Inference

For training the *canonicalization-enhanced encoder*, we take the same loss function as the KGE method adopted in the *query encoding* module. Taking the encoder built on ConvE as an example, the logistic sigmoid function σ is first applied to triple score ψ to obtain its ranking probability p_1:

$$p_1 = \sigma(\psi(e_h, r, e_t)) \tag{11}$$

Then the loss for triple (e_h, r, e_t) is calculated following Binary Cross Entropy (BCE):

$$loss_{(e_h,r,e_t)} = -(y \cdot log(p_1) + (1-y) \cdot log(1-p_1)) \tag{12}$$

where y is the label of (e_h, r, e_t), which is 1 if $(e_h, r, e_t) \in \mathcal{F}$ and 0 otherwise.

During the inference stage, the Top-K potential candidate tail NPs to query $(e_h, r, ?)$ are obtained by calculating the scores of triples consisting of $(e_h, r, ?)$ and all NPs.

3.4 Known Fact-Aware Re-Ranker

Prior CKG link prediction methods perform re-ranking with the query and predicted candidates as inputs, but fail to utilize canonical information of triples. To reduce triple sparsity in OpenKG, the *known fact-aware re-ranker* is proposed to re-score candidates by additionally exploiting query-related canonical triples for providing explicit contextualized hints.

We employ DistilBERT [Sanh *et al.*, 2019] followed by a classification head, which is a two-layer fully-connected neural network, to implement re-ranking, namely model $\mathcal{M}$. For query $(e_h, r, ?)$ and one of its candidate tail NPs e_t, they are concatenated into a query triple sentence $x_{q,t}$, i.e., $x_{q,t} = [x^{e_h}, x^r, x^{e_t}]$. Instead of using the query triple sentence alone for prediction, we adopt canonical triples $N_{triple}(e_h, r, ?)$ obtained in Eq. (6) to help re-score by concatenating them into a known factual sentence x_v, i.e., $x_v = [v_1; \ldots; v_{M_{triple}}]$. Then it is combined with the query triple sentence $x_{q,t}$ as an input pair fed into $\mathcal{M}$:

$$p_2 = \sigma(\mathcal{M}(x_{q,t}, x_v)) \tag{13}$$

where p_2 is the re-ranking probability of triple (e_h, r, e_t).

Model Training and Inference

For each query $(e_h, r, ?)$ in the training set, we use the ground-truth tail NPs to form the sentence pairs as positive samples, and extract the Top-M_{neg} negative candidate tail NPs predicted by *canonicalization-enhanced encoder* to form negative samples, where M_{neg} is a hyperparameter. During training, we filter out the query itself from the retrieved results. The *known fact-aware re-ranker* is fine-tuned by minimizing the BCE loss of p_2 and its label as in Eq. (12).

At inference stage, we re-rank Top-K candidate tail NPs through ensembling scores produced from *canonicalization-enhanced encoder* and *known fact-aware re-ranker*. For query $(e_h, r, ?)$, the ranking probability p_1 for one of its Top-K candidate tail NPs e_t is first scaled to p'_1 by the maximum ranking probability among the Top-K candidates:

$$p'_1 = \frac{p_1}{\max p_1^{(1:K)}} \tag{14}$$

where $p_1^{(1:K)}$ is the ranking probabilities of the Top-K candidates for query $(e_h, r, ?)$. At last, the final score $\hat{p}(e_h, r, e_t)$ for (e_h, r, e_t) is merged as:

$$\hat{p}(e_h, r, e_t) = \alpha p'_1 + (1 - \alpha) p_2 \tag{15}$$

where $\alpha \in [0, 1]$ is a score composition weight.

4 Experimental Setup

4.1 Dataset

We evaluate our approach on ReVerb20K and ReVerb45K datasets following CaRe [Gupta *et al.*, 2019], which are created through ReVerb Open KB [Fader *et al.*, 2011]. Details of these datasets are shown in Tab. 1, where "Gold NP Clusters" is the gold canonical clusters of NPs extracted through Freebase entity linking information [Gabrilovich *et al.*, 2013]. Inverse RPs are introduced for both datasets by adding the phrase "inverse of" to each RP. For more details, we refer readers to the related papers.

Dataset	ReVerb20K	ReVerb45K
#NPs	11,065	27,008
#RPs	11,058	21,623
#Gold NP Clusters	10,897	18,626
#Average NPs Per Cluster	1.02	1.45
#Train	15,499	35,970
#Validation	1,550	3,598
#Test	2,325	5,395

Table 1: Dataset statistics.

Dataset	d	K	M_{rp}	M_{triple}	M_{neg}	α
ReVerb20K	768	10	5	3	10	0.5
ReVerb45K	768	10	10	3	3	0.3

Table 2: Optimal values of hyperparameters.

4.2 Experimental Settings

We use canonical clusters of NPs provided in CaRe [Gupta *et al.*, 2019]. The pre-trained all-mpnet-base-v2 model is applied for f_{SBERT} in *sparsity reduction for RP and triple*. The pre-trained BERT-base-uncased model is employed for f_{BERT} in *pre-trained semantic embedding*. The pre-trained distilbert-base-uncased model is utilized to implement $\mathcal{M}$ in *known fact-aware re-ranker*. Note that we experiment with multiple pre-trained models and choose the above settings because of their good performance and high efficiency. We set the embedding dimension $d = 768$, and the number of re-ranking tail NPs $K = 10$. The value of M_{rp}, M_{triple}, and M_{neg} are all selected from $\{1, 3, 5, 7, 10\}$. The optimal settings of hyperparameters for both datasets are listed in Tab. 2.

For training the *canonicalization-enhanced encoder* (Sec. 3.3), we use the self-adaptive optimization method Adam [Kingma and Ba, 2014]. Here we report the training settings of the best-performing CEKFA[ResNet] framework, which will be introduced later. Both datasets use a batch size of 128 and an initial learning rate of 0.0001 with a scheduler that reduces the learning rate by a decay factor of 0.5 when the value of MRR has stopped improving for 2 epochs. The model is trained for a maximum of 500 epochs, which would be terminated if the MRR metric of the validation set has not improved within 10 epochs.

After training the *canonicalization-enhanced encoder* (Sec. 3.3), we freeze its parameters and then fine-tune the *known fact-aware re-ranker* (Sec. 3.4) for a maximum of 5 epochs with a batch size of 16, where the best model is chosen based on the accuracy of validation set. We employ AdamW [Loshchilov and Hutter, 2017] optimizer using a linear decay learning rate scheduler with a weight decay of 0.01 and a linear warm-up with 10% of the training data, where the initial learning rate is 2×10^{-5}.

Model	ReVerb20K					ReVerb45K				
	MRR	MR	Hits@1	Hits@3	Hits@10	MRR	MR	Hits@1	Hits@3	Hits@10
TransE [Bordes *et al.*, 2013]	0.138	1150.5	0.034	0.201	0.316	0.202	1889.5	0.122	0.243	0.346
ComplEx [Trouillon *et al.*, 2016]	0.038	4486.5	0.017	0.043	0.071	0.068	5659.8	0.054	0.071	0.093
R-GCN[‡] [Schlichtkrull *et al.*, 2018]	0.122	1204.3	-	-	0.187	0.042	2866.8	-	-	0.046
ConvE [Dettmers *et al.*, 2018]	0.262	1483.7	0.203	0.287	0.371	0.218	3306.8	0.166	0.243	0.314
KG-BERT [Yao *et al.*, 2019]	0.047	420.4	0.014	0.039	0.105	0.123	1325.8	0.070	0.131	0.223
RotatE [Sun *et al.*, 2019]	0.065	2861.5	0.043	0.069	0.108	0.141	3033.4	0.110	0.147	0.196
SpacESS [Nayyeri *et al.*, 2020]	0.206	2577.9	0.162	0.229	0.292	0.141	3144.0	0.106	0.149	0.205
PairRE [Chao *et al.*, 2021]	0.213	1366.2	0.166	0.229	0.296	0.205	2608.4	0.153	0.228	0.302
ResNet [Lovelace *et al.*, 2021]	0.224	2258.4	0.188	0.240	0.292	0.181	3928.9	0.150	0.196	0.242
BertResNet-ReRank [Lovelace *et al.*, 2021]	0.272	1245.6	0.225	0.294	0.347	0.208	2773.4	0.166	0.227	0.281
CaRe[†] [Gupta *et al.*, 2019]	0.318	973.2	-	-	0.439	0.324	1308.0	-	-	0.456
OKGIT[†] [Chandrahas and Talukdar, 2021]	0.359	527.1	0.282	0.394	0.499	0.332	773.9	0.261	0.363	0.464
OKGSE[†] [Xie *et al.*, 2022]	0.372	487.3	0.291	0.408	<u>0.524</u>	0.342	771.1	<u>0.274</u>	0.371	0.473
CEKFA[TransE]	0.316	795.0	0.249	0.346	0.425	0.265	1460.1	0.204	0.296	0.365
CEKFA[ComplEx]	0.328	512.0	0.265	0.359	0.433	0.303	1203.4	0.239	0.332	0.422
CEKFA[ConvE]	<u>0.386</u>	<u>383.1</u>	<u>0.307</u>	<u>0.423</u>	**0.525**	<u>0.354</u>	**630.9**	0.273	<u>0.392</u>	**0.504**
CEKFA[RotatE]	0.339	575.4	0.264	0.373	0.463	0.297	1111.1	0.216	0.340	0.447
CEKFA[PairRE]	0.378	**358.0**	0.301	0.415	0.516	0.330	<u>766.0</u>	0.247	0.373	0.483
CEKFA[ResNet]	**0.387**	416.7	**0.310**	**0.427**	0.515	**0.369**	884.5	**0.294**	**0.409**	<u>0.502</u>

Table 3: Link prediction results on OpenKG datasets. Best results are in bold and second best results are underlined. [†]: results are taken from corresponding original paper. [‡]: results are taken from [Gupta *et al.*, 2019].

At inference stage, Top-10 candidate tail NPs predicted by the *canonicalization-enhanced encoder* are re-scored by *known fact-aware re-ranker*, and the score composition weight α is picked from [0,1] that achieves the best MRR on validation set with an increment of 0.1. All experiments are conducted in Pytorch and on one 16G Tesla V100 GPU.

4.3 Baselines

We implement CEKFA with five typical KGE methods in CKG, including TransE [Bordes *et al.*, 2013], ComplEx [Trouillon *et al.*, 2016], ConvE [Dettmers *et al.*, 2018], RotatE [Sun *et al.*, 2019], and PairRE [Chao *et al.*, 2021], denoted as "CEKFA[*]". We also adopt ResNet [He *et al.*, 2016] following [Lovelace *et al.*, 2021] as the query encoding module to implement CEKFA, notated as "CEKFA[ResNet]".

We compare CEKFA with: (1) KGE methods in CKG: TransE [Bordes *et al.*, 2013], ComplEx [Trouillon *et al.*, 2016], R-GCN [Schlichtkrull *et al.*, 2018], ConvE [Dettmers *et al.*, 2018], KG-BERT [Yao *et al.*, 2019], RotatE [Sun *et al.*, 2019], SpacESS [Nayyeri *et al.*, 2020], PairRE [Chao *et al.*, 2021], the two-stage method [Lovelace *et al.*, 2021] (denoted as BertResNet-ReRank), and the query encoding module in it (denoted as ResNet); (2) KGE methods in OpenKG: CaRe [Gupta *et al.*, 2019], OKGIT [Chandrahas and Talukdar, 2021], and OKGSE [Xie *et al.*, 2022].

4.4 Evaluation Metrics

We evaluate CEKFA on the link prediction task following CaRe [Gupta *et al.*, 2019], which evaluates the rank of canonical clusters of NPs under filtered setting [Bordes *et al.*, 2013]. Standard ranking metrics are utilized: Mean Rank (MR), Mean Reciprocal Rank (MRR), Hits at 1 (Hits@1), Hits at 3 (Hits@3), and Hits at 10 (Hits@10).

5 Experimental Results

5.1 Main Results

Link prediction results of CEKFA against various baselines are shown in Tab. 3. CEKFA[ResNet] outperforms all baselines on MRR, Hits@1, and Hits@3, indicating that it performs best on accurate metrics. Compared to methods in OpenKG (the second field in Tab. 3), CEKFA[ResNet] achieves absolute improvements of 1.5% and 2.7% for MRR, 1.9% and 2.0% for Hits@1 on ReVerb20K and ReVerb45K respectively, which confirms the effectiveness of canonicalizing RPs and triples for OpenKG link prediction.

Unsurprisingly, most KGE methods in CKG show relatively poor performance when applied directly to OpenKG (the first field in Tab. 3), with the best MRR and Hits@1 on ReVerb20K and ReVerb45K implemented by BertResNet-ReRank and ConvE, respectively. Notably, KG-BERT performs moderately in MRR and Hits@1, but well in MR metric, with a similar phenomenon observed when applied to CKG [Wang *et al.*, 2021]. This is mainly because KG-BERT only encodes the textual knowledge of triples and neglects to learn the structural information, which makes it good at selecting NPs/entities with the right semantics as candidates, but unable to accurately predict the target owing to ambiguity problem. Therefore, when N is slightly large, it achieves good Hits@N (i.e., Top-N recall), but performs poorly for small N, resulting in poor MRR and good MR.

Comparing the first and third fields in Tab. 3, it is clear that the results of KGE models are improved by different degrees after being integrated with CEKFA. Interestingly, after applying CEKFA, ResNet obtains greater improvements than ConvE and achieves the best results, with absolute improvements of 16.3% and 18.8% for MRR, 12.2% and 14.4% for

Model	ReVerb20K					ReVerb45K				
	MRR	MR	Hits@1	Hits@3	Hits@10	MRR	MR	Hits@1	Hits@3	Hits@10
CEKFA[TransE]	**0.316**	**795.0**	**0.249**	**0.346**	**0.425**	**0.265**	**1460.1**	**0.204**	**0.296**	**0.365**
-KFARe	0.285	795.1	0.208	0.318	**0.425**	0.227	1460.3	0.149	0.268	**0.365**
-KFARe-RPCE	0.234	1034.1	0.159	0.264	0.372	0.211	1517.4	0.146	0.241	0.330
CEKFA[ComplEx]	**0.328**	**512.0**	**0.265**	**0.359**	**0.433**	**0.303**	**1203.4**	**0.239**	**0.332**	**0.422**
-KFARe	0.302	**512.0**	0.232	0.330	**0.433**	0.284	**1203.4**	0.210	0.321	**0.422**
-KFARe-RPCE	0.292	525.8	0.221	0.320	0.425	0.278	1242.6	0.207	0.310	0.410
CEKFA[ConvE]	**0.386**	**383.1**	**0.307**	**0.423**	**0.525**	**0.354**	**630.9**	**0.273**	**0.392**	**0.504**
-KFARe	0.370	387.6	0.290	0.407	**0.525**	0.338	631.0	0.256	0.374	**0.504**
-KFARe-RPCE	0.369	468.5	0.290	0.404	0.516	0.324	689.3	0.243	0.358	0.484
CEKFA[RotatE]	**0.339**	575.4	**0.264**	**0.373**	**0.463**	**0.297**	1111.1	**0.216**	**0.340**	**0.447**
-KFARe	0.323	575.5	0.246	0.355	**0.463**	0.279	1111.2	0.188	0.333	**0.447**
-KFARe-RPCE	0.307	**529.4**	0.222	0.347	0.457	0.266	**1095.4**	0.172	0.321	0.437
CEKFA[PairRE]	**0.378**	**358.0**	**0.301**	**0.415**	**0.516**	**0.330**	**766.0**	**0.247**	**0.373**	**0.483**
-KFARe	0.360	**358.0**	0.275	0.400	**0.516**	0.316	**766.0**	0.223	0.366	**0.483**
-KFARe-RPCE	0.342	406.9	0.256	0.382	0.502	0.297	789.0	0.205	0.349	0.466
CEKFA[ResNet]	**0.387**	**416.7**	**0.310**	**0.427**	**0.515**	**0.369**	**884.5**	**0.294**	**0.409**	**0.502**
-KFARe	0.372	416.8	0.296	0.403	**0.515**	0.354	884.6	0.277	0.392	**0.502**
-KFARe-RPCE	0.362	441.1	0.279	0.398	0.509	0.333	892.2	0.255	0.370	0.484
CEKFA[ResNet]_TripleOnlyReranking	0.370	416.8	0.294	0.402	0.515	0.353	884.6	0.275	0.392	0.502
CEKFA[ResNet]_BM25Retrieving	0.383	416.7	0.307	0.420	0.515	0.361	884.6	0.285	0.402	0.502

Table 4: Link prediction results for ablation and variants of CEKFA. Best results for each field are in bold.

Hits@1 on ReVerb20K and ReVerb45K respectively. This indicates the effectiveness of employing deeper convolutional networks to capture the interactions between dense embeddings with rich semantics. Moreover, CEKFA obtains up to 29.0% and 23.5% absolute improvements for MRR metric on ReVerb20K and ReVerb45K (CEKFA[ComplEx]). Even with the smallest boost, CEKFA obtains 12.4% (CEKFA[ConvE]) and 6.3% (CEKFA[TransE]) in MRR on these two datasets respectively, suggesting the effectiveness and generalization ability of the proposed framework.

5.2 Ablation and Variants Study

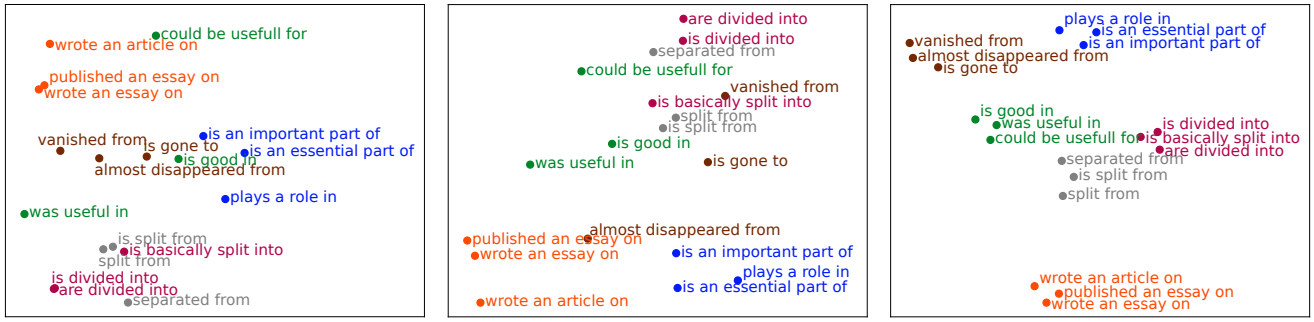
Ablation study of the proposed canonicalization methods in CEKFA is presented in Tab. 4. We find that: (1) For CEKFA[ResNet], removing the *known fact-aware re-ranker* (-KFARe) decreases the MRR by 1.5% on both datasets. And for CEKFA[*] constructed from different KGE methods, the removal of KFARe leads to maximum decreases of 3.1% and 3.8% for MRR, 4.1% and 5.5% for Hit@1 on ReVerb20K and ReVerb45K respectively, both of which are built on TransE. It illustrates the crucial role of triple canonicalization to share knowledge of related triples. (2) For CEKFA[ResNet]-KFARe, removing *RP canonicalization encoding* (-RPCE) further reduces the MRR by 1.0% and 2.1% on ReVerb20K and ReVerb45K. For CEKFA[*]-KFARe, the absence of RPCE leads to maximum reductions of 5.1% and 2.1% for MRR, 4.9% and 2.2% for Hit@1 on ReVerb20K and ReVerb45K, which are integrated with TransE and ResNet, respectively. This implies the effectiveness of RP canonicalization in sharing knowledge of similar RPs to reduce their high sparsity. Overall, the results show that leveraging RP and triple canonicalization helps to reduce their sparsity and enhance the performance of CKG methods.

In addition, to investigate the effectiveness of *known fact-aware re-ranker*, two variants are further developed. From the results shown at the bottom of Tab. 4, we can observe that: (1) Re-ranking by query triple sentence $x_{q,t}$ without known factual sentence x_v (CEKFA[ResNet]_TripleOnlyReranking) gives no gains, and even results in a drop of 0.1%∼0.2% in MRR on both datasets. It indicates that known facts composed of canonical triples are indispensable to the triple re-ranking stage which performs triple canonicalization and provides contextualized hints. (2) Retrieving relevant known facts by BM25 (CEKFA[ResNet]_BM25Retrieving) that learns sparse representations performs worse than by SBERT that learns dense representations. It is primarily owing to the ability of SBERT to better handle complex sentences with the same semantics but different expressions.

5.3 Sparsity Analyses

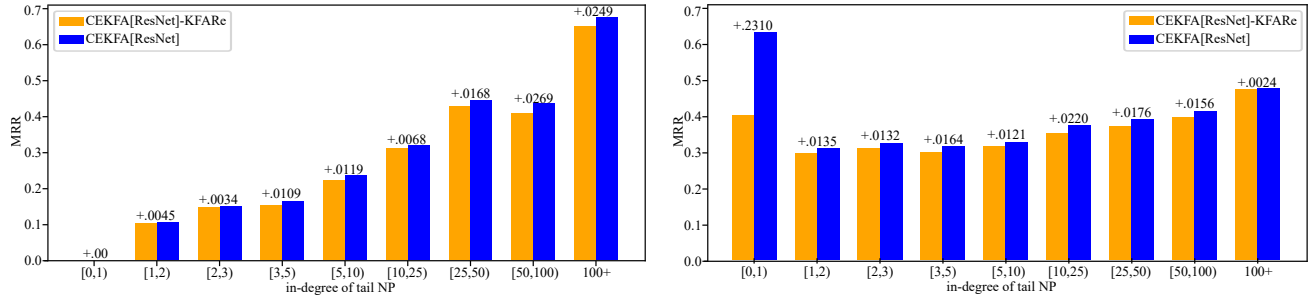
Sparsity of RPs

To intuitively understand the sparsity reduction of RPs, we compare the canonical neighbors of several randomly chosen RPs obtained from CEKFA[ResNet] with CESI [Vashishth *et al.*, 2018], which performs OpenKG canonicalization task. Results are displayed in Tab. 5, where only the Top-3 canonical neighbor RPs are reported for each RP. It can be seen that RP canonicalization obtained from CEKFA[ResNet] is semantically similar, rather than only superficially similar, e.g., the canonical neighbor RPs obtained for "was an inmate at". Besides, we obtain specific canonicalization for each RP instead of clustering them, which is flexible and highly fault-



(a) CaRe (b) OKGIT (c) CEKFA

Figure 3: Comparison of the t-SNE visualization about RP embeddings obtained by CaRe [Gupta *et al.*, 2019], OKGIT [Chandrahas and Talukdar, 2021] and our CEKFA[ResNet]. Among them, only CEKFA[ResNet] utilizes RP canonicalization to enhance embedding learning.



(a) ReVerb20K (b) ReVerb45K

Figure 4: Effects of re-ranking in CEKFA[ResNet] over different in-degree range of tail NPs. The less the in-degree, the sparser it is.

RPs	CESI	CEKFA[ResNet]
was an inmate at	was an engineer at was an exile in is an institution in	was imprisoned at was jailed in was imprisoned in
was jailed in	was arrested in is currently jailed in is arrested in	was arrested in was imprisoned in is currently jailed in
is not mentioned again in	is not really in is not just in is not only in	makes no mention of is no mention of fails to mention

Table 5: Canonical neighbor RPs obtained by CESI [Vashishth *et al.*, 2018] and CEKFA[ResNet]. Error canonical RPs are in red.

tolerant. For example, in CESI, "was an engineer at" is incorrectly clustered with "was an inmate at", resulting in all RPs in this cluster having wrong canonicalization information.

Furthermore, for comparison of the learned RP embeddings with and without canonicalization, we project the high-dimensional embedding to a 2-dimensional space using t-SNE [Van der Maaten and Hinton, 2008] for visualization. Owing to the lack of gold canonicalization of RPs, we randomly select six RPs with different semantics in experiments, and for each RP we manually choose two more RPs with the same semantics. Fig. 3 displays the results of CEKFA[ResNet] compared against CaRe [Gupta *et al.*, 2019] and OKGIT [Chandrahas and Talukdar, 2021], where RP canonicalization is not considered in CaRe and OKGIT. It can be seen that RPs with the same semantic are well clustered by CEKFA[ResNet]. Besides, CEKFA[ResNet] can distinguish RPs that have similar tokens but different semantics, e.g., "split from" and "is basically split into".

Sparsity of Triples

To visualize the effect of triple sparsity reduction for predicting NPs with different sparse levels, Fig. 4 shows the improvements brought by the *known fact-aware re-ranker* based on the in-degree of tail NPs in test triples. We find that the performance across all sparse levels of tail NPs can be improved by re-ranking with known facts to reduce the sparsity of query triples. Noticeably, re-ranking with triple canonicalization brings a significant performance boost of 23.10% in ReVerb45K when tail NPs are "extremely sparse", i.e., their in-degree in the training data is 0. It indicates that canonical triples can provide query-related contextualized hints, especially in predicting unseen NPs.

To analyze the improvements to *known fact-aware re-ranker* brought by known factual sentences consisting of canonical triples, we present in Tab. 6 the Top-3 candidate tail NPs predicted by CEKFA[ResNet] before and after re-ranking for several query pairs, as well as the Top-3 retrieved

query	gold tail NP	canonical triples	CEKFA[ResNet]-KFARe	CEKFA[ResNet]
(chi minh city, is the biggest city of, ?)	vietnam	chi minh city is the largest city in vietnam; chi minh city is also popularly known as saigon; chi minh city was the capital of south vietnam;	vietnamese **vietnam** china	**vietnam** vietnamese south vietnam
(phil gramm, went to work for, ?)	ub	phil gramm is also an officer of ub; gramm was married to wendy gramm; phil is the creator of pretty good privacy;	obama bush clinton	**ub** ub financial service inc obama
(simon, inverse of was called, ?)	peter	simon was growing to be simon peter; simon had mixed judaism; bartholomew was another name for nathanael;	jesus david archuleta archuleta	jesus **peter** joseph
(levitra, is available in, ?)	europe	levitra is taken with lortab; levitra is approved in europe; levitra is similar to viagra;	germany india china	germany **europe** china

Table 6: Comparison of tail NPs predicted by CEKFA[ResNet] before and after re-ranking. Target tail NPs are in bold.

Inference Time (#samples/s)	TransE	ComplEx	ResNet
baseline ([*])	239	257	643
CEKFA[*]-KFARe	211	245	568
Relative Change	**TransE**	**ComplEx**	**ResNet**
relative efficiency change	-11.72%	-4.67%	-10.53%
relative performance change	+12.37%	+276.47%	+95.58%

Table 7: Comparison of inference times, and relative changes in time and performance for three baseline methods before and after applying the *canonicalization-enhanced encoder* (denoted as [*] and CEKFA[*]-KFARe) on ReVerb45K dataset.

canonical triples. It can be found that known facts in the training data related or similar to the query pair can be retrieved as canonicalization, even if their tokens are superficially less similar, e.g., "simon was growing to be simon peter" can be retrieved for query *(simon, inverse of was called, ?)*. Moreover, better inferences can be made with the support of canonical triples. For instance, "phil gramm is also an officer of ub" provides contextual information for query *(phil gramm, went to work for, ?)*, which leads to correct prediction.

5.4 Time Complexity

We analyze the additional time complexity of CEKFA compared to baselines during the inference process in each stage. For *canonicalization-enhanced encoder*, the increased time complexity is $O(T(M_{np} + M_{rp}))$, coming from the canonicalization encoding of NP and RP in each triple, where T is the number of triples, M_{np} and M_{rp} are the average number of canonical neighbors for each NP and RP ($M_{np} < 2$, $M_{rp} \leq 10$). As for the *known fact-aware re-ranker*, the time complexity is $O(TK)$, stemming from encoding each query with one of its top-K candidate tails.

Tab. 7 compares the inference times of three baseline methods before and after applying the *canonicalization-enhanced encoder*, as well as the relative changes in time and performance. It can be seen that the *canonicalization-enhanced encoder* built on baseline method without re-ranking stage (CEKFA[*]-KFARe) achieves performance gains by 12.37%$\sim$276.47%, which only decreases the inference time by 4.67%$\sim$11.72%. Although the time cost increases, which comes from the canonicalization encoding of NP and RP, it is much less than the performance benefits brought by the *canonicalization-enhanced encoder*. As for *known fact-aware re-ranker* (KFARe), we compare its efficiency in CEKFA[ResNet] with the re-ranking stage of BertResNet-ReRank [Lovelace *et al.*, 2021], where the number of samples predicted per second on ReVerb45K is 61 and 151, respectively. The relatively slow re-ranking time in CEKFA is reasonable since we include the canonical triples as part of the input, which increases the encoding time.

6 Conclusion

In this paper, we propose a Canonicalization-Enhanced Known Fact-Aware (CEKFA) framework for OpenKG link prediction by reducing the sparsity of RPs and triples. For RP sparsity reduction, a similarity-driven RP canonicalization method is introduced to collect semantically similar RPs as canonical information. To reduce the sparsity of triples, a known fact-aware triple canonicalization method is developed by retrieving query-related known facts to provide contextualized hints. These two kinds of canonical knowledge are integrated into the canonicalization-enhanced encoder and known fact-aware re-ranker for link prediction. Experiments show that CEKFA outperforms previous baselines on two OpenKG datasets, ReVerb20K and ReVerb45K, and extensive analyses suggest the effectiveness and generalization ability of the proposed approach.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (Grant No. 62006243, Grant No. 62025208, and Grant No. 61932001), and the Xiangjiang Laboratory Foundation (Grant No. 22XJ01012).

References

[Bollacker *et al.*, 2008] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, page 1247–1250, New York, NY, USA, 2008. Association for Computing Machinery.

[Bordes *et al.*, 2013] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *Neural Information Processing Systems (NIPS)*, pages 1–9, 2013.

[Broscheit *et al.*, 2020] Samuel Broscheit, Kiril Gashteovski, Yanjie Wang, and Rainer Gemulla. Can we predict new facts with open knowledge graph embeddings? a benchmark for open link prediction. In *Annual Meeting of the Association for Computational Linguistics*, 2020.

[Chandrahas and Talukdar, 2021] . Chandrahas and Partha Talukdar. OKGIT: Open knowledge graph link prediction with implicit types. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2546–2559, Online, August 2021. Association for Computational Linguistics.

[Chao *et al.*, 2021] Linlin Chao, Jianshan He, Taifeng Wang, and Wei Chu. PairRE: Knowledge graph embeddings via paired relation vectors. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4360–4369, Online, August 2021. Association for Computational Linguistics.

[Cho *et al.*, 2014] Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Conference on Empirical Methods in Natural Language Processing*, 2014.

[Dash *et al.*, 2021] Sarthak Dash, Gaetano Rossiello, Nandana Mihindukulasooriya, Sugato Bagchi, and Alfio Gliozzo. Open knowledge graphs canonicalization using variational autoencoders. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10379–10394, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

[Dettmers *et al.*, 2018] Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. Convolutional 2d knowledge graph embeddings. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*, AAAI'18/IAAI'18/EAAI'18. AAAI Press, 2018.

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

[Dubey *et al.*, 2018] Mohnish Dubey, Debayan Banerjee, Debanjan Chaudhuri, and Jens Lehmann. Earl: Joint entity and relation linking for question answering over knowledge graphs. In Denny Vrandečić, Kalina Bontcheva, Mari Carmen Suárez-Figueroa, Valentina Presutti, Irene Celino, Marta Sabou, Lucie-Aimée Kaffee, and Elena Simperl, editors, *The Semantic Web – ISWC 2018*, pages 108–126, Cham, 2018. Springer International Publishing.

[Fader *et al.*, 2011] Anthony Fader, Stephen Soderland, and Oren Etzioni. Identifying relations for open information extraction. In *Proceedings of the 2011 conference on empirical methods in natural language processing*, pages 1535–1545, 2011.

[Gabrilovich *et al.*, 2013] Evgeniy Gabrilovich, Michael Ringgaard, and Amarnag Subramanya. Facc1: Freebase annotation of clueweb corpora, version 1 (release date 2013-06-26, format version 1, correction level 0). *Note: http://lemurproject. org/clueweb09/FACC1/Cited by*, 5:140, 2013.

[Gu *et al.*, 2018] Jiatao Gu, Yong Wang, Kyunghyun Cho, and Victor OK Li. Search engine guided neural machine translation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[Gupta *et al.*, 2019] Swapnil Gupta, Sreyash Kenkre, and Partha Talukdar. Care: Open knowledge graph embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 378–388, 2019.

[He *et al.*, 2016] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[Hochreiter and Schmidhuber, 1997] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[Khandelwal *et al.*, 2020] Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. Generalization through memorization: Nearest neighbor language models. In *International Conference on Learning Representations*, 2020.

[Kingma and Ba, 2014] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *Computer Science*, 2014.

[Kolluru *et al.*, 2021] Keshav Kolluru, Mayank Singh Chauhan, Yatin Nandwani, Parag Singla, et al. Cear:

Cross-entity aware reranker for knowledge base completion. *arXiv preprint arXiv:2104.08741*, 2021.

[Lin and Chen, 2019] Xueling Lin and Lei Chen. Canonicalization of open knowledge bases with side information from the source text. In *2019 IEEE 35th International Conference on Data Engineering (ICDE)*, pages 950–961, 2019.

[Lin *et al.*, 2015] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. Learning entity and relation embeddings for knowledge graph completion. In *Twenty-ninth AAAI conference on artificial intelligence*, 2015.

[Liu *et al.*, 2021] Yinan Liu, Wei Shen, Yuanfei Wang, Jianyong Wang, Zhenglu Yang, and Xiaojie Yuan. Joint open knowledge base canonicalization and linking. In *Proceedings of the 2021 International Conference on Management of Data*, SIGMOD '21, page 2253–2261, New York, NY, USA, 2021. Association for Computing Machinery.

[Loshchilov and Hutter, 2017] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2017.

[Lovelace *et al.*, 2021] Justin Lovelace, Denis R. Newman-Griffis, Shikhar Vashishth, Jill Fain Lehman, and Carolyn Penstein Ros'e. Robust knowledge graph completion with stacked convolutions and a student re-ranking network. *Proceedings of the conference. Association for Computational Linguistics. Meeting*, 2021:1016–1029, 2021.

[Nayyeri *et al.*, 2020] Mojtaba Nayyeri, Chengjin Xu, Sahar Vahdati, Nadezhda Vassilyeva, Emanuel Sallinger, Hamed Shariat Yazdi, and Jens Lehmann. Fantastic knowledge graph embeddings and how to find the right space for them. In *The Semantic Web – ISWC 2020: 19th International Semantic Web Conference, Athens, Greece, November 2–6, 2020, Proceedings, Part I*, page 438–455, Berlin, Heidelberg, 2020. Springer-Verlag.

[Nickel *et al.*, 2016] Maximilian Nickel, Lorenzo Rosasco, and Tomaso Poggio. Holographic embeddings of knowledge graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, 2016.

[Pennington *et al.*, 2014] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[Reimers and Gurevych, 2019] Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China, November 2019. Association for Computational Linguistics.

[Robertson and Zaragoza, 2009] Stephen Robertson and Hugo Zaragoza. *The probabilistic relevance framework: BM25 and beyond*. Now Publishers Inc, 2009.

[Saha and others, 2018] Swarnadeep Saha et al. Open information extraction from conjunctive sentences. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2288–2299, 2018.

[Sanh *et al.*, 2019] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. In *NeurIPS EMC^2 Workshop*, 2019.

[Schlichtkrull *et al.*, 2018] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. In *European semantic web conference*, pages 593–607. Springer, 2018.

[Sun *et al.*, 2019] Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. Rotate: Knowledge graph embedding by relational rotation in complex space. In *International Conference on Learning Representations*, 2019.

[Trouillon *et al.*, 2016] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. Complex embeddings for simple link prediction. In *International Conference on Machine Learning*, pages 2071–2080. PMLR, 2016.

[Van der Maaten and Hinton, 2008] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

[Vashishth *et al.*, 2018] Shikhar Vashishth, Prince Jain, and Partha Talukdar. Cesi: Canonicalizing open knowledge bases using embeddings and side information. In *Proceedings of the 2018 World Wide Web Conference*, pages 1317–1327, 2018.

[Wang *et al.*, 2021] Bo Wang, Tao Shen, Guodong Long, Tianyi Zhou, Ying Wang, and Yi Chang. Structure-augmented text representation learning for efficient knowledge graph completion. In *Proceedings of the Web Conference 2021*, pages 1737–1748, 2021.

[Wang *et al.*, 2022] Shuo Wang, Yichong Xu, Yuwei Fang, Yang Liu, S. Sun, Ruochen Xu, Chenguang Zhu, and Michael Zeng. Training data is more valuable than you think: A simple and effective method by retrieving from training data. In *Annual Meeting of the Association for Computational Linguistics*, 2022.

[Xie *et al.*, 2022] Tingyu Xie, Peng Peng, Hongwei Wang, and Yusheng Liu. Open knowledge graph link prediction with segmented embedding. In *2022 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, 2022.

[Yao *et al.*, 2019] Liang Yao, Chengsheng Mao, and Yuan Luo. Kg-bert: Bert for knowledge graph completion. *arXiv preprint arXiv:1909.03193*, 2019.

[Zhang *et al.*, 2019] Shuai Zhang, Yi Tay, Lina Yao, and Qi Liu. Quaternion knowledge graph embeddings. *Advances in neural information processing systems*, 32, 2019.