Commonsense Knowledge Graph Completion Via Contrastive Pretraining and Node Clustering

Siwei Wu, Xiangqing Shen, and Rui Xia[*]
School of Computer Science and Engineering,
Nanjing University of Science and Technology, China
{wusiwei, xiangqing.shen, rxia}@njust.edu.cn

Abstract

The nodes in the commonsense knowledge graph (CSKG) are normally represented by free-form short text (e.g., word or phrase). Different nodes may represent the same concept. This leads to the problems of edge sparsity and node redundancy, which challenges CSKG representation and completion. On the one hand, edge sparsity limits the performance of graph representation learning; On the other hand, node redundancy makes different nodes corresponding to the same concept have inconsistent relations with other nodes. To address the two problems, we propose a new CSKG completion framework based on Contrastive Pretraining and Node Clustering (CPNC). Contrastive Pretraining constructs positive and negative head-tail node pairs on CSKG and utilizes contrastive learning to obtain better semantic node representation. Node Clustering aggregates nodes with the same concept into a latent concept, assisting the task of CSKG completion. We evaluate our CPNC approach on two CSKG completion benchmarks (CN-100K and ATOMIC), where CPNC outperforms the state-of-the-art methods. Extensive experiments demonstrate that both Contrastive Pretraining and Node Clustering can significantly improve the performance of CSKG completion. The source code of CPNC is publicly available on https://github.com/NUSTM/CPNC.

1 Introduction

Commonsense knowledge graphs (CSKG) have been widely used to build commonsense-grounded AI applications, such as question answering (Lv et al., 2020), visual question answering (Zhu et al., 2020), sentiment analysis (Li et al., 2022), dialogue system (Tu et al., 2022), etc. Commonsense knowledge graphs such as ConceptNet (Speer et al., 2017) and ATOMIC (Sap et al., 2019) provide a structured way of representing a commonsense concept, which consists of a head node, a tail node, and the relation edge. Nodes in commonsense knowledge graphs are typically represented by free-form short text (word or phrase), resulting in many different nodes representing the same concept. Figure 1 shows a subgraph of ConceptNet, where nodes in the same color indicate the same concept. E.g., "*have breakfast*", "*take breakfast*" and "*eat breakfast*" all express the concept of "***eat breakfast***". This problem also results in a large number of missing edges between nodes, as illustrated by the dashed lines in Figure 1. E.g., "*eat breakfast*" and "*prepare food*" have a "*HasPrerequisite*" relation, but such relation is missing between "*eat breakfast*" and "*cook food*".

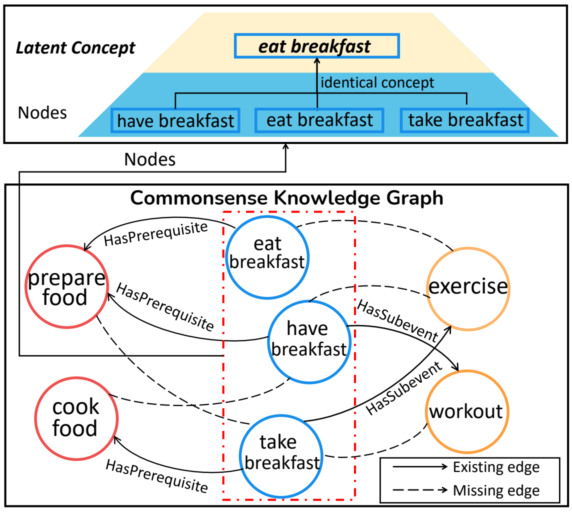

Figure 1: Illustration of a subgraph and the latent concept in ConceptNet. The nodes of the same color belong to the same latent concept.

On the one hand, node redundancy in CSKG makes different nodes of the same concept have inconsistent relations with other nodes. In CSKG representation and completion, it would be beneficial to make use of the latent concept information behind different nodes to help learn more semantic-general representations. However, this intuition was ignored by most of the existing work in

[*] Corresponding author

Findings of the Association for Computational Linguistics: ACL 2023, pages 13977–13989
July 9-14, 2023 ©2023 Association for Computational Linguistics

CSKG completion. On the other hand, as analyzed by Malaviya et al. (2020), edge sparsity in CSKG limits information propagation in graph neural networks. Upon graph neural networks, researchers further incorporated pre-trained language model such as BERT to enhance the semantic representation of nodes (Malaviya et al., 2020; Ju et al., 2022; Wang et al., 2021). However, fine-tuning BERT is still imperfect in representing the commonsense knowledge graph, which is made up of linked nodes represented in a free-form short text.

To tackle the two issues, we propose a new CSKG completion framework based on the Encoder-Decoder architecture, which contains two core modules ***C**ontrastive **P**retraining and **N**ode **C**lustering* (CPNC). Contrastive Pretraining is to alleviate the difficulty of node representation learning induced by edge sparsity. Through contrastive learning on positive and negative head-tail node pairs, the embedding distance between related nodes becomes closer, and that between unrelated nodes becomes farther so as to learn better node representations. Node Clustering aims to address the edge inconsistency issue caused by node redundancy. We cluster nodes with close semantic representations and take the mean vector as the latent concept representation for nodes in this cluster. To assist CSKG completion, the latent concept representation is fused with the node representation.

We evaluate our CPNC framework on two CSKG completion benchmarks, i.e., CN-100K and ATOMIC. The results show that our model outperforms the state-of-the-art models for this task significantly. Ablation studies demonstrate that both Contrastive Pretraining and Node Clustering modules can significantly improve the performance of CSKG completion. Further experiments verify that our model can consistently improve as the sparsity of knowledge graphs increases: the higher the sparsity, the more improvement our model achieves.

2 Related Work

We briefly review traditional knowledge graph completion, and then pay more attention to commonsense knowledge graph completion.

Traditional Knowledge Graph Completion Many knowledge graph completion methods have been proposed, which can be classified into three types: embedding-based completion methods, path-finding-based completion methods, and logical rule-based completion methods. The embedding-based method learned the relation and node embedding by shortening the distance between the head-relation pair representation and the tail node representation, which had good scalability. Moreover, convolution has been proven to be an effective operation for acquiring the head-relation representation (Dettmers et al., 2018; Shang et al., 2019) in the embedding-based method. The path-based methods used the random walk inference based algorithm to find the related path. They achieved the prediction of missing tuples by comparing the related path with the relation to be predicted (Lao and Cohen, 2010; Khot et al., 2011). The rule-based methods utilized the induction rules to simplify the path-finding process in the knowledge graphs (Ren et al., 2020; Yang and Song, 2020). We refer the reader to (Ji et al., 2022) for more details about knowledge graph completion.

Commonsense Knowledge Graph Completion The existing methods assume specific relations, dense edges, and sufficient training samples in the knowledge graph. However, the sparsity of edges and the abstract nature of relations pose challenges for directly applying these methods to the CSKG.

Early methods in this field employed a strategy where, for a given head and tail, all relations were used to generate a large number of tuples. BiLSTMs and linear transformations were then utilized to score these tuples, allowing for the prediction of missing tuples in the CSKG (Saito et al., 2018; Li et al., 2016; Jastrzebski et al., 2018). Building upon this foundation, Shen et al. (2022) took a step further. They normalized the tail nodes and used RoBERTa to score these tuples. This approach resulted in the curation of Dense-ATOMIC, a CSKG with increased coverage and a richer set of multi-hop paths.

these methods did not take into account structural information and may suffer from computational inefficiency during inference. Another line of studies attempted to use the translated-based approach to reduce computational consumption (Malaviya et al., 2020; Ju et al., 2022; Wang et al., 2021). Furthermore, they incorporated GCN for extracting graph representations and fine-tuned BERT in the CSKG to obtain semantic representations. This integration capitalizes on the complementary nature of these two representations. In addition, these methods have applied convolutional layers to enhance the performance of CSKG completion. This further improves the effectiveness of the com-

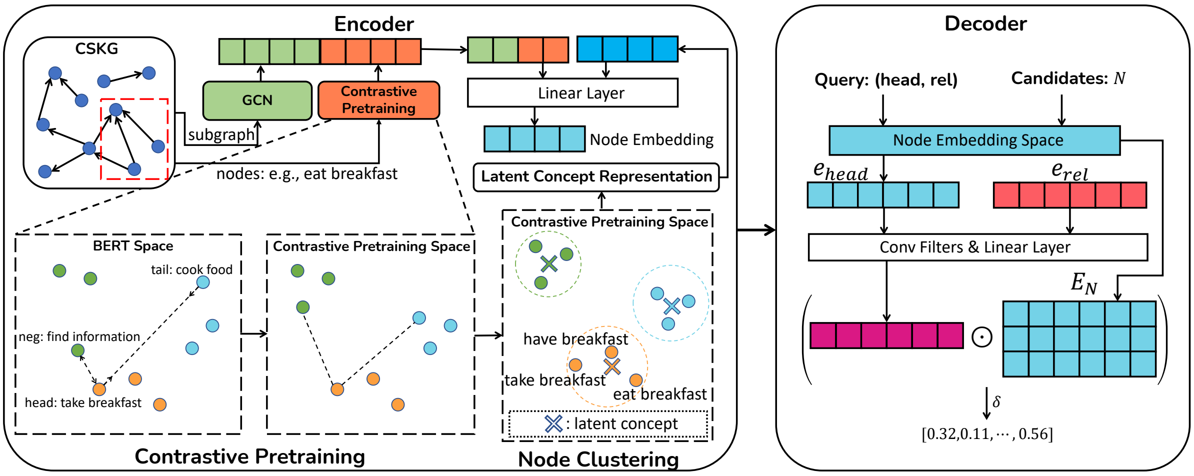

Figure 2: Overall architecture of CPNC. The Encoder is to get node embedding. Harnessing the power of GCN and Contractive Pretraining, the Encoder transforms CSKG into rich semantic and structural representations. In detail, the lower left section utilizes the mechanism of Contrastive Pretraining. Then, we describe the node clustering process in the lower right, driven by semantic representations from Contractive Pretraining. Finally, the fusion of semantic, structure, and latent concepts in the upper right yields node embeddings. The Decoder part is to rank the nodes in candidates set by Node Embedding.

pletion task. However, it was observed that simply fine-tuning BERT (Devlin et al., 2019) with pretraining tasks is hard to effectively capture the sentence-level semantic connection between two nodes. In response to this limitation, Su et al. (2022) proposed MICO, which learns distinct node representations for head nodes under varying relations, proving evidence that high-quality node representation can significantly aid CSKG completion. Moreover, different nodes in the CSKG may express the same concept. The node redundancy will bring difficulties to the commonsense inference on the CSKG (Jung et al., 2022), while the previous CSKG completion methods ignored.

Compared to these methods, we address the issue of node redundancy by employing a clustering algorithm. Our work stands out by learning the semantic representations of nodes through sentence-level semantic connections and integrating latent concept information into node representations.

3 Task Definition

Given a CSKG $G = (N, V, R)$ where N is the set of nodes, V is the set of edges and R is the set of relations. We regard each tuple $v_i = (h, rel, t)$ in the CSKG as a sample, which is composed of head node h, tail node t and relation rel, where $h, t \in N$, $v_i \in V$, and $rel \in R$. Given a query (h, rel) formed by a head node h and a relation rel, the target of the CSKG completion is to maximize the score of the tail node t. Following the previous work (Malaviya et al., 2020), for an edge (h, rel, t) existing in the CSKG, we also add an inverse edge (t, rel^{-1}, h) to the graph, where rel^{-1} is the inverse relation of rel.

4 Approach

In this paper, we propose a new framework, CPNC, for CSKG completion using an Encoder-Decoder architecture (see Figure 2). The Encoder incorporates semantic, graph structure, and latent concept representation obtained from Contrastive Pretraining, Graph Convolutional Network (GCN), and Node Clustering, respectively, to acquire node representations. Given query (h, rel), the Decoder ranks the nodes in the candidate set N and finds tail nodes for the query by the rank.

4.1 Contrastive Pretraining

The current mainstream approach for completing CSKG utilizes translation-based methods and relies on GCN to represent the graph structure of nodes. Although GCNs are effective in modeling graph structure, they have limitations in capturing graph structure information in CSKG due to sparse edges. To address this issue, these methods incorporate semantic information by fine-tuning BERT on CSKG using Masked Language Model (MLM) and Next Sentence Prediction (NSP) tasks.

However, those methods are imperfect in modeling CSKGs composed of nodes presented as short text. MLM, which is a token-level task, focuses on modeling the connection between masked words and other words, making it unsuitable for achieving sentence-level node representation. Similarly, NSP, while capable of modeling the semantic connection between two nodes at the sentence level, suffers from a mismatch between the input of the pretraining phase and the CSKG completion phase. During pretraining, NSP requires a pair of head and tail nodes for prediction, but during CSKG completion, only a single node is used for sentence representation. Consequently, NSP is not well-suited for the task of learning node representations, and similar observations have been made in the field of sentence representation learning (Li et al., 2020).

To obtain better node representations for the CSKG completion, we introduce Contrastive Pretraining (CP), a new method that sufficiently leverages semantic information at the sentence level. CP focuses on CSKG completion and differs from MICO (Su et al., 2022) by not incorporating relation categories. Instead, it fine-tunes BERT's node representation using contrastive learning and capture sentence-level connections between nodes.

4.1.1 Building Contrastive Learning Samples

Assuming node pairs linked in the CSKG are semantically related, we consider node pairs without a link in the CSKG as semantically unrelated.

For each arbitrary edge (h, rel, t) in CSKG, we randomly sample a node $\bar{t}$ having no linking with the head node h as its hard negative tail and construct a contrastive learning training sample $c = (h, t, \bar{t})$. As shown in Figure 3, "take breakfast" and "cook food" are the head and tail nodes in a edge of CSKG. We randomly select a node "*find information*" having no linking with "*take breakfast*" to construct a training sample *("take breakfast", "cook food", "find information").*

The previous methods primarily focused on the head nodes while constructing negative samples, neglecting the importance of capturing unrelated semantics among the tail node set. To tackle this problem, we additionally constructed a contrastive learning sample $c_{reverse} = (t, h, \bar{h})$ for the tail node t in the edge (h, rel, t).

4.1.2 Multiple Negatives Ranking Loss

We employ BERT for node embedding by inputting all nodes in a batch separately and applying Mean Pooling on the final layer's output. Then, we use $x(n)$ to denote the representation of a specific node n obtained through BERT.

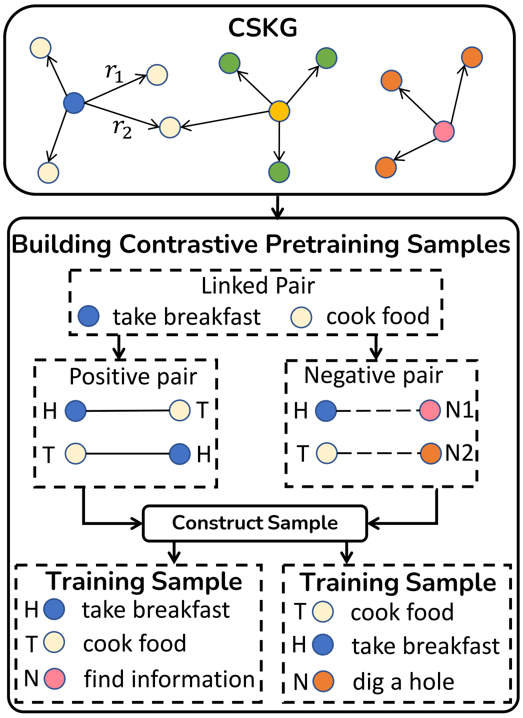

Figure 3: Main process of building Constractive Pretraining samples.

In order to ensure that semantically related nodes are positioned closely in the embedding space, we utilize contrastive learning and employ the Multiple Negatives Ranking Loss for contrastive learning.

For a given sample $c_i = (h_i, t_i, \bar{t}_i)$ in a batch, we employ the Multiple Negatives Ranking Loss. We consider h_i paired with t_i as the positive sample while treating h_i paired with other tail nodes t_j in the batch as negative samples. Additionally, we consider h_i paired with all hard negative tails in the batch as negative samples as well. The objective is to minimize the semantic distance between the nodes in positive samples within the batch. The formulation of the multiple negatives ranking loss is as follows:

$$L = -\sum_{i=1}^{M} \log \frac{e^{D(h_i,t_i))}}{\sum_{j=1}^{M} e^{D(h_i,t_j)} + e^{D(h_i,\bar{t}_j)}}, \quad (1)$$

where $D(h, t) = (x(h) \cdot x(t)^T)/(\|x(h)\| \cdot \|x(t)\|)$ represents the cosine similarity of two nodes, M is the number of samples in the a batch.

4.2 Encoder

The Encoder aims to generate node representations for CSKG completion,as illustrated in the left part of Figure 2.

The node representation comprises two components,

1) the semantic representation matrix of nodes E_{sem} obtained through Contrastive Pretraining:

$$E_{sem} = \text{Contrastive-Pretraining}(N), \tag{2}$$

2) the graph structure representation matrix of nodes acquired using a Graph Convolutional Network (GCN),

$$E_{graph} = \text{GCN}(G). \tag{3}$$

Specifically, e_{sem} and e_{graph} represent the semantic and graph representations of a particular node, respectively, which can be derived from E_{sem} and E_{graph}.

4.3 Node Clustering

On the basis of the semantic node representation e_{sem}, we use the K-means algorithm to cluster all nodes N in the graph as follows:

$$\{\mathcal{C}_1, \mathcal{C}_2, ..., \mathcal{C}_K\} = \text{K-means}(N), \tag{4}$$

where $\mathcal{C}_i$ is the i-th cluster, and K represents the number of clusters. Nodes in $\mathcal{C}_i$ are considered to have the same latent concept. As shown in the Node Clustering part of Figure 2, "*eat breakfast* ", "*have breakfast* " and "*take breakfast* " are in the same cluster and have the same latent concept.

We define the representation of the latent concept as the mean of semantic representation of all nodes in a cluster, denoted as mean$(\mathcal{C}_k)$, where $\mathcal{C}_k$ contains a set of nodes in the same cluster.

To acquire the final node representation, we combine the node semantic representation, e_{sem}, the graph representation, e_{graph}, and the latent concept representation, mean$(\mathcal{C}_k)$. This fusion is performed as follows:

$$e = [e_{sem}; e_{graph}; \text{mean}(\mathcal{C}_k)] \cdot W_{embedding}, \tag{5}$$

where $W_{embedding}$ is a weight matrix. The resulting node embeddings are considered as the **node embedding space**.

Intuitively, Node Clustering enhances CSKG completion by leveraging information from nodes within the same latent concept.

We also use a progressive masking process, following Malaviya et al. (2020), to integrate latent concept information. Initially, the latent concept representation is fully masked, and throughout the first 100 training epochs, it is gradually unmasked, improving the overall performance.

4.4 Decoder

The goal of Decoder is to rank the nodes in the candidate set, which is considered as N in our work.

In order to obtain the query representation, we first use a convolutional layer to fuse the relation and head node representation:

$$q = \text{Conv}(e_h, e_{rel}), \tag{6}$$

where e_h is the representation of the head nodes in node embedding space obtained from Eq. 5, e_{rel} is the representation of relation rel and Conv is a convolution operation.

Then, we predict a 0-1 distribution vector, based on the representation of the query q, indicating the likelihood of each node in the candidate set N becoming the tail node:

$$p_t = \delta(qW_{conv}E_N), \tag{7}$$

where W_{conv} is a weight matrix, E_N is the representation matrix of candidate nodes in node embedding space obtained from Eq. 5, and δ is a sigmoid function.

5 Experiments

5.1 Experimental Setup

5.1.1 Datasets

We evaluate our CPNC framework on two CSKG completion benchmarks, i.e., CN-100K (Speer et al., 2017) and ATOMIC (Sap et al., 2019).

CN-100K is a dataset that encompasses general commonsense knowledge. This version contains 36 relation types and the Open Mind Common Sense (OMCS) entries from ConceptNet (Speer and Havasi, 2013). The average length of the nodes in CN-100K is 2.85 words. Following Malaviya et al. (2020)'s split, the training set contains 10,000 tuples, and the validation set and test set both contain 1200 tuples.

ATOMIC is an atlas of everyday commonsense reasoning and primarily focuses on event-level commonsense knowledge in the form of if-then relations. It comprises 9 relation types, with an average of 4.40 words per node. We split ATOMIC following Malaviya et al. (2020)'s work, where the training set consists of 610,536 tuples, while the validation and test sets contain 87,700 tuples and 87,701 tuples, respectively.

Methods		CN-100K				ATOMIC			
Type	Model	MRR	HITS@1	@3	@10	MRR	HITS@1	@3	@10
KG-adapted	DistMult	8.97	4.51	9.76	17.44	12.39	9.24	15.18	18.30
	ComplEx	11.40	7.42	12.45	19.01	14.24	**13.27**	14.13	15.96
	ConvE	20.88	13.97	22.91	34.02	10.07	8.24	10.29	13.37
	ConvTransE	18.68	7.87	23.87	38.95	12.94	12.92	12.95	12.98
Generation-based	COMeT-Normalized	6.07	0.08	2.92	21.17	3.36	0.00	2.15	15.75
	COMeT-Total	6.21	0.00	0.00	24.00	4.91	0.00	2.40	21.60
CSKG-dedicated	RGAT	43.97	30.75	51.54	69.34	-	-	-	-
	SGBC	49.12*	37.71*	56.67*	71.29*	10.25*	8.72*	10.54*	13.26*
	InductivE	56.92*	45.54*	63.38*	78.63*	13.19*	10.26*	13.61*	18.83*
Ours	CPNC-S	54.52	45.33	61.46	75.92	13.14	10.11	13.75	18.80
	CPNC-I	**59.00**	**48.29**	**65.04**	**79.13**	**14.38**	10.53	**15.22**	**21.79**

Table 1: CSKG completion results on CN-100K and ATOMIC. The result of HITS@1 was not reported in Wang et al. (2021). To fairly compare our method with the previous method in the same setting, we rerun the code of InductivE and SGBC on ATOMIC and CN-100K and mark the results of rerun experiments with *. Those results are a little lower than that in the original paper. For KG-adapted and Generation methods, we reuse the results reported in Malaviya et al. (2020).

5.1.2 Evaluation Metric

We evaluate the performance of our method using MRR and HITS, following previous CSKG completion methods (Wang et al., 2021; Malaviya et al., 2020; Ju et al., 2022). The results are reported by averaging over both forward tuples (h, rel, t) and inverse tuples (t, rel^{-1}, h). Moreover, because the nodes in CSKG are represented in free-form text, it is possible for nodes other than the golden tail nodes to be considered reasonable tail nodes. To address this, we conduct a human evaluation to assess the predictions made by our models.

5.1.3 Implementation Details

To perform Contrastive Pretraining, we use the contrastive learning framework provided in `https://github.com/UKPLab/sentence-transformers`. Our approach employs BERT-large as the base model, which contains 340M parameters. During training, we utilized a batch size of 128 and conducted training for 3 epochs on the CN-100K and ATOMIC datasets. We choose the Adam optimizer for optimization, setting the learning rate to 1e-4 for BERT-large and 5e-5 for the MLP. For the remaining hyperparameters, we used the default values provided by the framework.

For the CPNC-I and CPNC-S model, we use experimental settings proposed by Malaviya et al. (2020) and Wang et al. (2021), respectively. Both models are trained for a minimum of 200 epochs using BERT-large, which consists of 340M parameters, to encode semantic representations. During training, we evaluate the MRR on the development set every 10 epochs for CN-100K and ATOMIC. Training continues until no further improvement in MRR is observed. We select the model checkpoint that achieves the highest MRR on the development set for testing.

5.2 Compared Systems

We compare our approach with nine baseline systems across three categories.

5.2.1 KG-adapted Methods

We adapt classic knowledge graph completion methods for CSKG completion. **DistMult** (Yang et al., 2015) employed a bi-linear product to calculate score of a tuple; **ComplE** (Trouillon et al., 2016) utilized imaginary number representation to effectively handle a large number of relations in the knowledge graph; **ConvE** (Dettmers et al., 2018) fused the representation of the source node and the relation through a 2D convolution layer to obtain the representation of query $(\mathrm{h}, \mathrm{rel})$; **ConvTransE** extended ConvE by incorporating the translational properties of TransE.

5.2.2 Generation-based Methods

COMeT (Bosselut et al., 2019) is a Transformer-based knowledge generation model. Following Malaviya et al. (2020), we adapt COMeT for the CSKG completion and only evaluate in the forward direction. Additionally, we also use their modifications to the ranking method used in COMeT. Specifically, **COMET-Normalized** and **COMET-Total** use the normalized negative log-likelihood scores and total log-likelihood scores, respectively, to rank nodes in the candidate set.

5.2.3 CSKG-dedicated Methods

SIM+GCN+BERT+ConvTransE (**SGBC**) densified the CSKG by connecting the synthetic edges between similar semantic nodes, improving the graph structure representation (Malaviya et al., 2020). Moreover, they used fine-tuned BERT to encode the semantic information of the nodes.

InductivE (Wang et al., 2021) is proposed to enhance the unseen entity representation with neighboring structural information by densifying Graph. Relational graph attention networks (**RGAT**) are proposed weighted the importance of neighbor nodes of each node to obtain a better node representation.

5.2.4 Our CPNC Methods

By incorporating CPNC with two CSKG-dedicated methods (SGBC and InductivE), we obtain two models, i.e., **CPNC-S** and **CPNC-I**.

5.3 Main Results

In Table 1, we report experimental results of KG-adapted methods, Generation methods, CSKG-dedicated methods and our methods on CN-100K and ATOMIC.

On CN-100K, both KG-adapted methods and Generation methods exhibit unsatisfactory performance, with MRR values below 21% and HITS@1 values below 14%. These results demonstrate that directly applying KG-adapted and Generation methods to CSKG completion is ineffective. In contrast, CSKG-dedicated methods achieve significantly better results on CN-100K, with MRR values above 43% and HITS@1 values above 30%. These metrics are twice as high as those obtained by KG-adapted and Generation methods, highlighting the advantage of CSKG-dedicated methods. Comparing our proposed CPNC-S and CPNC-I models with mainstream CSKG-dedicated methods (SGBC and InductivE), we observe performance improvements of 5.40% and 2.08% on MRR, respectively. Notably, our CPNC-I model sets a new state-of-the-art result on CN-100K.

On ATOMIC, we draw a similar conclusion. The KG-adapted methods exhibit poor performance, while CSKG-dedicated methods achieve better results. In addition, compared with the SGBC and InductivE models, our CPNC-S and CPNC-I models outperform the SGBC and InductivE models by 2.89% and 1.19% on MRR, respectively. And the CPNC-I model achieves the state-of-the-art result on ATOMIC.

We conduct Paired t-Test and the result proves that the improvement of CPNC is significant.

5.4 Ablation Study

		MRR	HITS@10
CN-100K	**CPNC-S**	**54.52**	**75.92**
	-w/o CP	51.90	76.08
	-w/o NC	49.74	71.38
	SGBC	49.12	71.29
	CPNC-I	**59.00**	**79.13**
	-w/o CP	57.16	74.90
	-w/o NC	58.30	78.75
	InductivE	56.92	78.63
ATOMIC	**CPNC-S**	**13.14**	**18.80**
	-w/o CP	12.77	17.15
	-w/o NC	12.46	17.72
	SGBC	10.25	13.26
	CPNC-I	**14.38**	**21.79**
	-w/o CP	13.22	19.16
	-w/o NC	14.21	21.38
	InductivE	13.19	18.83

Table 2: MRR and HITS@10 of removing CP and NC from CPNC on CN-100K and ATOMIC.

To demonstrate the effectiveness of Contrastive Pretraining (CP) and Node Clustering (NC) on CSKG completion, we perform the ablation study where we removed CP and NC. The results are presented in Table 2.

When NC was removed, we can observe that the performance of both CPNC-S and CPNC-I decreases on CN-100K and ATOMIC. This indicates that NC's latent concept representation effectively assists CSKG completion. It is worth noting that CP can still bring improvement compared with the previous method, showing that the semantic representation provided by CP is crucial for CSKG completion.

CPNC, which combines NC and CP, consistently yields the highest result. By replacing CP with finetuned BERT (Malaviya et al., 2020), the performance of both CPNC-S and CPNC-I drops. However, they still achieve comparable results with the previous CSKG-dedicated methods. This demonstrates that the CP requires a high-quality semantic representation of nodes to acquire good latent concept representation.

5.5 Human Evaluation

For a given query (h, rel), additional nodes besides the golden tail node could also become reasonable tail nodes. For example, given a query "*(do housework, Causes)* ", besides golden tail node " *clean the house*", " *house get clean*" and " *clean the room*" are also reasonable tail nodes. However, automated metrics fail to cover these nodes. To address this issue, we randomly select 200 queries from the test set of the CN-100K and ATOMIC. For each query, we use our methods and current mainstream methods to rank the nodes in the candidate set, We manually calculate the number of reasonable tail nodes in the top 10 candidates. The results are shown in Table 3.

	ATOMIC	CN-100K
CPNC-I	**70.22**	**83.20**
InductivE	64.45	76.40
CPNC-S	60.23	69.40
SGBC	62.11	65.40

Table 3: Average percentage of reasonable tuples by human evaluation results.

Observing human evaluation results, the CPNC model has an excellent performance in CSKG completion. Besides, our method is significantly higher than the current mainstream methods under the human evaluation results, which further validates the effectiveness of our approach.

5.6 Model Adaptability

The CPNC can be effectively applied to various methods. In our experiment, we incorporate CPNC into SGBC and InductivE, resulting in significant improvements.

The SGBC method initializes the input of GCN randomly and combines graph structural representation acquired by GCN and semantic representation obtained by BERT as the node representation; InductivE initializes the input of GCN with BERT and uses the output of GCN as the node representation. Despite the differences in their structures, both methods benefit from the application of CP and NC.

On CN-100K, CPNC-S and CPNC-I bring improvements of 5.40% and 2.08% in MRR, respectively; On ATOMIC, CPNC-S and CPNC-I yield MRR improvements of 2.89% and 1.19%, respectively. This result demonstrates the adaptability of our method.

5.7 Discussion on the Effect of Relieving Sparsity

To evaluate the effectiveness of CPNC in mitigating the impact of edge sparsity in the CSKG, we conducted a comparative study between CPNC-S and SGBC on CN-100K with varying sparsity levels. The removal of edges in the CSKG results in increased sparsity, with a higher number of deleted edges corresponding to a higher sparsity degree.

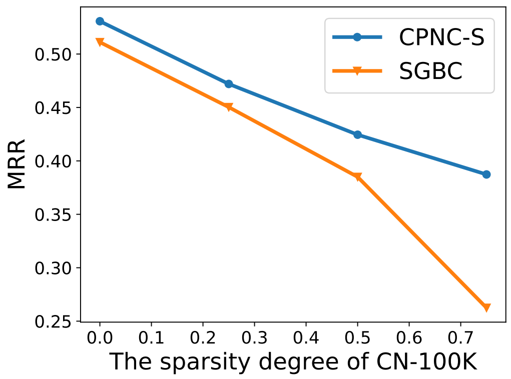

Figure 4: CSKG completion results of SGBC and CPNC-S on CN-100K with various degrees of sparsity.

As shown in Figure 4, when the sparsity degree increases, CSKG completion performance deteriorates. CPNC-S consistently outperforms SGBC in terms of MRR on CN-100K across different sparsity levels, indicating CPNC's ability to mitigate edge sparsity in the CSKG. CPNC-S achieves a 12.48% and 3.95% increase in MRR compared to SGBC at sparsity extents of 25% and 50% respectively. These results demonstrate that CPNC is more effective in alleviating edge sparsity in the CSKG, with greater improvements observed for sparser graphs.

5.8 Discussion on the Number of Latent Concepts

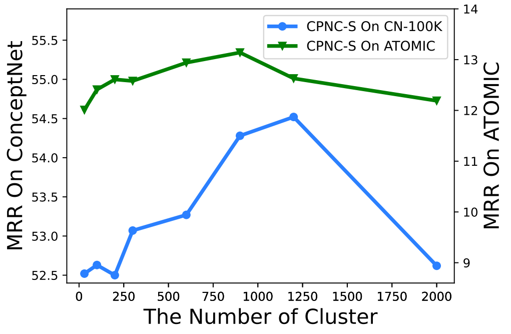

Figure 5: CSKG completion performance with different cluster numbers on CN-100K and ATOMIC.

Different numbers of clusters can lead to varying levels of granularity in latent concepts, which in turn can impact the performance of CSKG completion. In order to investigate this, we conducted experiments with the different numbers of clusters on CN-100K and ATOMIC datasets and present the results in Figure 5.

On CN-100K, we observed an overall increasing trend in completion results as the number of clusters increased. However, the rate of improvement tends to diminish after reaching 900 clusters. The maximum MRR achieved when using the number of clusters is 1200.

On ATOMIC, the MRR initially increases with an increasing number of clusters but starts to decline once the number of clusters reaches 900.

Based on our main results, we selected 1200 clusters for CN-100K and 900 clusters for ATOMIC, allowing us to effectively capture latent concepts through cluster nodes.

5.9 Discussing on Contrastive Learning Loss

In order to obtain a better semantic representation of nodes, we experiment with five kinds of contrastive learning methods and observe their performance in CSKG completion. The result is shown in Table 4.

Methods	MRR
Contrastive Loss	40.03
MTriplet Loss	44.36
MICO	45.93
Batch Semi Hard Triplet Loss	27.92
Multiple Negatives Ranking Loss	**49.74**

Table 4: CSKG completion performance of CPNC-S using different Contrastive Learning Loss on CN-100K.

In order to compare those contrastive learning methods, we use the node embedding obtained by those methods to complete CSKG without node clustering. Each contrastive learning method yields distinct results, and our main result selects Multiple Negatives Ranking Loss due to its highest MRR.

6 Conclusion

In this work, we propose a new CSKG completion framework CPNC to address issues arising from node redundancy and edge sparsity. CPNC obtains better semantic node information through Contrastive Pretraining, which alleviates the problems caused by edge sparsity. CPNC also utilizes the latent concept representation acquired through Node Clustering to alleviate the problem caused by node redundancy. On CN-100K and ATOMIC, experimental results and extensive analysis demonstrate the effectiveness of Contrastive Pretraining and Node Clustering.

Limitations

Due to the limitation of time and resources, in this work, we select a relatively small number of clusters during the clustering process, which results in coarse-grained clustering. Fine-grained clustering can provide a better latent concept but will also lead to increased computational resources and time consumption. We will attempt to trade-off between the cost and the granularity of clustering in future work to further explore the impact of the latent concept on CSKG completion. Besides, our Node Clustering module is not integrated in an end-to-end manner in our work; We will consider using the topic neural network to construct an end-to-end model.

Ethics Statement

We would like to thank Malaviya et al. (2020) and Wang et al. (2021) for their code on commonsense knowledge graph completion. Their models are licensed under MIT, which allows copying, modifying, merging, publishing, and distributing of the material.

In the stage of human evaluation, we employed three graduate students experienced in natural language processing for human evaluation. We paid the graduate students about $8 per hour, well above the local average wage, and engaged in constructive discussions if they had concerns about the process.

Acknowledgements

This work was supported by the Natural Science Foundation of China (No. 62076133), and the Natural Science Foundation of Jiangsu Province for Distinguished Young Scholars (No. BK20200018).

References

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi. 2019. COMET: commonsense transformers for automatic knowledge graph construction. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 4762–4779. Association for Computational Linguistics.

Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Convolutional 2d knowledge graph embeddings. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence,* *(AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 1811–1818. AAAI Press.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. 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, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Stanislaw Jastrzebski, Dzmitry Bahdanau, Seyedarian Hosseini, Michael Noukhovitch, Yoshua Bengio, and Jackie Chi Kit Cheung. 2018. Commonsense mining as knowledge base completion? A study on the impact of novelty. *CoRR*, abs/1804.09259.

Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and Philip S. Yu. 2022. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE Trans. Neural Networks Learn. Syst.*, 33(2):494–514.

Jinhao Ju, Deqing Yang, and Jingping Liu. 2022. Commonsense knowledge base completion with relational graph attention network and pre-trained language model. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management, Atlanta, GA, USA, October 17-21, 2022*, pages 4104–4108. ACM.

Yong-Ho Jung, Jun-Hyung Park, Joon-Young Choi, Mingyu Lee, Junho Kim, Kang-Min Kim, and SangKeun Lee. 2022. Learning from missing relations: Contrastive learning with commonsense knowledge graphs for commonsense inference. In *Findings of the Association for Computational Linguistics: ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 1514–1523. Association for Computational Linguistics.

Tushar Khot, Sriraam Natarajan, Kristian Kersting, and Jude W. Shavlik. 2011. Learning markov logic networks via functional gradient boosting. In *11th IEEE International Conference on Data Mining, ICDM 2011, Vancouver, BC, Canada, December 11-14, 2011*, pages 320–329. IEEE Computer Society.

Ni Lao and William W. Cohen. 2010. Relational retrieval using a combination of path-constrained random walks. *Mach. Learn.*, 81(1):53–67.

Bohan Li, Hao Zhou, Junxian He, Mingxuan Wang, Yiming Yang, and Lei Li. 2020. On the sentence embeddings from pre-trained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 9119–9130. Association for Computational Linguistics.

Jiangnan Li, Fandong Meng, Zheng Lin, Rui Liu, Peng Fu, Yanan Cao, Weiping Wang, and Jie Zhou. 2022. Neutral utterances are also causes: Enhancing conversational causal emotion entailment with social commonsense knowledge. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI 2022, Vienna, Austria, 23-29 July 2022*, pages 4209–4215. ijcai.org.

Xiang Li, Aynaz Taheri, Lifu Tu, and Kevin Gimpel. 2016. Commonsense knowledge base completion. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*. The Association for Computer Linguistics.

Shangwen Lv, Daya Guo, Jingjing Xu, Duyu Tang, Nan Duan, Ming Gong, Linjun Shou, Daxin Jiang, Guihong Cao, and Songlin Hu. 2020. Graph-based reasoning over heterogeneous external knowledge for commonsense question answering. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8449–8456. AAAI Press.

Chaitanya Malaviya, Chandra Bhagavatula, Antoine Bosselut, and Yejin Choi. 2020. Commonsense knowledge base completion with structural and semantic context. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 2925–2933. AAAI Press.

Hongyu Ren, Weihua Hu, and Jure Leskovec. 2020. Query2box: Reasoning over knowledge graphs in vector space using box embeddings. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Itsumi Saito, Kyosuke Nishida, Hisako Asano, and Junji Tomita. 2018. Commonsense knowledge base completion and generation. In *Proceedings of the 22nd Conference on Computational Natural Language Learning, CoNLL 2018, Brussels, Belgium, October 31 - November 1, 2018*, pages 141–150. Association for Computational Linguistics.

Maarten Sap, Ronan Le Bras, Emily Allaway, Chandra Bhagavatula, Nicholas Lourie, Hannah Rashkin, Brendan Roof, Noah A. Smith, and Yejin Choi. 2019. ATOMIC: an atlas of machine commonsense for if-then reasoning. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA,* *January 27 - February 1, 2019*, pages 3027–3035. AAAI Press.

Chao Shang, Yun Tang, Jing Huang, Jinbo Bi, Xiaodong He, and Bowen Zhou. 2019. End-to-end structure-aware convolutional networks for knowledge base completion. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 3060–3067. AAAI Press.

Xiangqing Shen, Siwei Wu, and Rui Xia. 2022. Dense-atomic: Construction of densely-connected and multi-hop commonsense knowledge graph upon ATOMIC. *CoRR*, abs/2210.07621.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 4444–4451. AAAI Press.

Robyn Speer and Catherine Havasi. 2013. Conceptnet 5: A large semantic network for relational knowledge. In Iryna Gurevych and Jungi Kim, editors, *The People's Web Meets NLP, Collaboratively Constructed Language Resources*, Theory and Applications of Natural Language Processing, pages 161–176. Springer.

Ying Su, Zihao Wang, Tianqing Fang, Hongming Zhang, Yangqiu Song, and Tong Zhang. 2022. MICO: A multi-alternative contrastive learning framework for commonsense knowledge representation. *CoRR*, abs/2210.07570.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 2071–2080. JMLR.org.

Quan Tu, Yanran Li, Jianwei Cui, Bin Wang, Ji-Rong Wen, and Rui Yan. 2022. MISC: A mixed strategy-aware model integrating COMET for emotional support conversation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 308–319. Association for Computational Linguistics.

Bin Wang, Guangtao Wang, Jing Huang, Jiaxuan You, Jure Leskovec, and C.-C. Jay Kuo. 2021. Inductive learning on commonsense knowledge graph completion. In *International Joint Conference on Neural Networks, IJCNN 2021, Shenzhen, China, July 18-22, 2021*, pages 1–8. IEEE.

Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding entities and relations for learning and inference in knowledge bases. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Yuan Yang and Le Song. 2020. Learn to explain efficiently via neural logic inductive learning. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Zihao Zhu, Jing Yu, Yujing Wang, Yajing Sun, Yue Hu, and Qi Wu. 2020. Mucko: Multi-layer cross-modal knowledge reasoning for fact-based visual question answering. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pages 1097–1103. ijcai.org.

ACL 2023 Responsible NLP Checklist

A For every submission:

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

□ 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 and section 1 Introduction

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

B ☑ Did you use or create scientific artifacts?

Section 5.1.3 Implementation Details

☑ B1. Did you cite the creators of artifacts you used?
Section 5.1.3 Implementation Details

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

☑ 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 5.1.3 Implementation Details

☒ 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?
We use publically available datasets, and the authors of the dataset have made the corresponding declaration.

☒ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
We will provide those data after the review process.

☑ 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.
5.1.1 section Datasets

C ☑ Did you run computational experiments?

5.1.3 section Implementation Details

☑ 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 5.1.3 Implementation Details

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 5.1.3 Implementation Details

☑ 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?
5.3 Main Results

☑ 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.)?
Section 5.1.3 Implementation Details

D ☑ Did you use human annotators (e.g., crowdworkers) or research with human participants?
3

☑ 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.?
We perform simple human annotation and evaluation, there is no need of providing the full text.

☑ 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)?
Ethics Statement section

☒ 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?
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?
Ethics Statement section