Amplifying Diversity and Quality in Commonsense Knowledge Graph Completion (Student Abstract)

Liu Yu, Fenghui Tian, Ping Kuang [*], Fan Zhou

University of Electronic Science and Technology of China, Chengdu, Sichuan 610054, China
liu.yu@std.uestc.edu.cn, greyhuhu@std.uestc.edu.cn, kuangping@uestc.edu.cn, fan.zhou@uestc.edu.cn

Abstract

Conventional commonsense knowledge graph completion (CKGC) methods provide inadequate sequence when fine-tuning or generating stages and incorporate *full* fine-tuning, which fail to align with the autoregressive model's pre-training patterns and have insufficient parameter efficiency. Moreover, decoding through beam or greedy search produces low diversity and high similarity in generated tail entities. Hence, we resort to prefix-tuning and propose a lightweight, effective pipeline to enhance the quality and diversity of extracted commonsense knowledge. Precisely, we measure head entity similarity to yield and then concatenate top-k tuples before each target tuple for prefix-tuning the source LM, thereby improving the efficiency and speed for pre-trained models; then, we design a penalty-tailored diverse beam search (p-DBS) for decoding tail entities, producing a greater quantity and diversity of generated commonsense tuples; besides, a filter strategy is utilized to filter out invalid commonsense knowledge. Through extensive automatic evaluations, including ChatGPT scoring, our method can extract diverse, novel, and accurate commonsense knowledge (CK).

Introduction

Recent attempts to utilize LMs for commonsense knowledge graph completion are mainly divided into two categories: (1) Fine-tuning based methods, which aim to adapt the LM representations acquired through pre-training by adding novel nodes and edges into the existing seed knowledge base (KB). COMET (Bosselut et al. 2019) adapts GPT (Radford et al. 2018) to perform tail entities completion given head entities and relations based on the practical KB. Such CKGC methods are based on KB to construct the dataset (fine-tuning stage) and the devised prompt text (inference stage). (2) Direct prompting methods: Recent efforts attempt to extract implicit or explicit knowledge by querying the LMs with well-designed prompts. For instance, (Petroni et al. 2019) prompting LMs by "*Dante was born in* __" for the implicit unstructured answer "*Florence*". The latest work (Wang et al. 2023) is a Chinese KB explicitly derived from GPT-3.5, a resource-intensive process that heavily relies on the capabilities of LMs.

[*]Corresponding author.

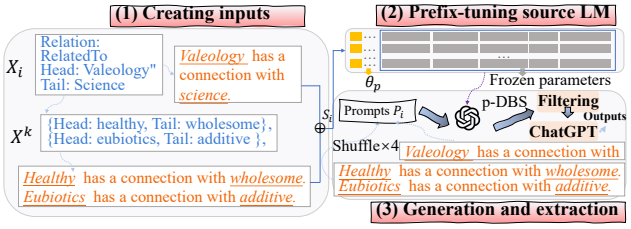

Figure 1: Our lightweight and effective CKGC pipeline.

Current methods show promise but come with significant drawbacks: (1) Fine-tuning methods involving *full* parameters in LMs can be time-consuming and inefficient. Also, such methods typically input a short knowledge triple into the LM, whether finetuning or generating stage, which fails to match the adequate sequence pre-training models. Furthermore, their generated knowledge exhibits limited diversity and high similarity, as they employ beam or greedy search for decoding. (2) Methods directly prompting extracted implicit knowledge lack key properties like accessibility, easy browsing, and editing. While capturing explicit knowledge is feasible, it heavily depends on LMs' strong in-context and few-shot learning abilities, making it unsuitable for other robust LMs like GPT2-xl. In this work, we present a lightweight, effective pipeline (in Fig. 1) to amplify the quality and diversity of CKGC. Specifically, we first concatenate top-k similar tuples-based sentences for prefix-tuning the source LM, boosting training efficiency. Moreover, we design a penalty-tailored diverse beam search (p-DBS) for decoding the tail entities, generating a larger and more varied set of commonsense tuples; besides, we use a filter module to filter out invalid commonsense knowledge.

Method

Creating inputs and prefix-tuning the source LM. We first transform the pre-defined relations into sentence templates (e.g., a relation "RelatedTo" is with the template of " [H] has a connection with [T]"), where [H]/[T] represents the head/tail entity. For each tuple $X_i = (H_i, R_i, T_i)$ in KB, we match top-k similar head entities within the same R_i via:

$$sim(h_i, h_j) = \frac{h_i \cdot h_j}{\|h_i\|\|h_j\|}$$

where h denotes the vector representations of head entity H. This matching yield k most similar tuples (denoted as $X^k = \{X_i\}_1^k$). Next, we convert X^k and X_i into natural language sentences using pre-defined relation templates and concatenate them, with commas serving as separators, and "." is appended at the end to form the input S_i to X_i. The input construction method helps the model learn both the knowledge distribution in the KB and accurate sentence-breaking rules, aligning with the long sentence patterns of pre-training source LM. Then, we proceed with prefix-tuning the source LM using the constructed dataset until it converges. Instead of fine-tuning the full LM, we provide a set of continuous trainable prefixes θ_p before the LM's parameters (keeping it frozen) as extra hints for optimization.

Knowledge generation via e-DBS and extraction. For each X_i, we first obtain its top-k similar tuples X^k in the same way of creating inputs; next, we obtain head entity H_i and its relation R_i without T_i (denoted as $\overline{X}_i$) for pending generating new ones. We concatenate X^k and $\overline{X}_i$, with commas serving as separators, as the prompting text P_i.

When prompting the prefix-tuned LM for generation, a penalty-tailored diverse beam search (p-DBS) algorithm is devised to enable generation diversity. Specifically, as we treat the sequence preceding truncation symbols as tail entities, however, due to the penalty mechanism, if the preceding group has generated a sentence break symbol, the current group may not break sentences, resulting in a valid tail entity. To solve this, we modify DBS to alleviate penalties for characters involved in sentence breaks to get p-DBS via:

$$penalty_i = \alpha freq_i$$

where $\alpha \in (0, 1)$ is the penalty coefficient, and $freq_i$ is the frequency of the i-th penalty character. We employ p-DBS for tail entities decoding and obtain the generated tail entity T_g, and (H_i, R_i, T_g) is the acquired knowledge.

Filter strategies. Although we perform prefix-tune on the source LM, enabling it to learn the knowledge distribution from KB. However, generating incorrect or invalid knowledge is still inevitable, so a filter strategy is deemed essential. The beam search score of each T_g is saved by:

$$\begin{aligned} S_{tail} =& \frac{1}{L} \log P\left(y_1, \ldots, y_L \mid \mathbf{c}\right) \\ =& \frac{1}{L} \sum_{t'=1}^{L} \log P\left(y_{t'} \mid y_1, \ldots, y_{t'-1}, \mathbf{c}\right) \end{aligned}$$

where L denotes the max length of newly generated tokens y, $\mathbf{c}$ is the prompt context. S_{tail} measures the quality of T_g. In practice, we enhance the confidence of S_{tail} through shuffling prompt order, generating multiple tail entities, and aggregating the identical instances scores. If $S_{tail} < \beta$, the corresponding T_g is subject to distillation.

Experiment

We use ConceptNet as knowledge seeds, and GPT2-xl as a source LMs. We set num_beam and num_beam_groups as 20 of p-DBS to force differences between groups. k is a random number between 3-5. We compare with three baselines: **LAMA**, **COMET**, and **BertNet** (Hao et al. 2023). Table 1 reports the overall accuracy, novelty and quantity of the generated CK. Note that $(\cdot)$ is the accuracy of newly generated CK. We report average score of our generated tuples scoring with GPT-3.5 API (closer to 100 is better) by providing several scoring examples to the system.

Model	Accuracy	Novelty	Quantity	Score
LAMA	0.60(P@10)	-	-	-
COMET	0.66	0.72	220K	-
BertNet	0.55	0.87	220K	-
Pre-train	0.29 (0.30)	0.88	223K	34.47
Fine-tune	0.48 (0.49)	**0.89**	**384K**	52.56
Ours	**0.85 (0.85)**	**0.89**	319K	**69.45**

Table 1: Statistics of KGs completion with various methods. - denotes no report in the original paper.

Compared to baselines, our method achieves large-scale (384K/319K before and after filtration) and high-novelty (0.89) while keeping the accuracy (both 0.85 for overall and new), proving the effectiveness of our proposed p-DBS and filtering strategy. LAMA and COMET enable smaller quantities, as they merely produce a piece of CK for each $\overline{X}_i$ via beam or greedy search. Moreover, they choose the highest probability word as the answer, which usually already exists in KB, leading to lower novelty. Although BertNet incorporates new relations, it obtains lower accuracy due to its sole use of an LM as the source of knowledge without any external KB, which is a reasonable outcome.

Acknowledgments

This work was supported in part by Key R&D Projects of Sichuan Provincial Science and Technology Plan (Grant No. 2023YFG0114), Central University Basic Research Business Fee Support Project (Grant No. ZYGX2021YGLH220), and Chengdu Key R&D Support Plan (Grant No. 2021YF0800019GX).

References

Bosselut, A.; Rashkin, H.; Sap, M.; Malaviya, C.; Celikyilmaz, A.; and Choi, Y. 2019. COMET: Commonsense transformers for automatic knowledge graph construction. *ACL*.

Hao, S.; Tan, B.; Tang, K.; Ni, B.; Shao, X.; Zhang, H.; Xing, E.; and Hu, Z. 2023. BertNet: Harvesting Knowledge Graphs with Arbitrary Relations from Pretrained Language Models. In *ACL*.

Petroni, F.; Rocktäschel, T.; Lewis, P.; Bakhtin, A.; Wu, Y.; Miller, A. H.; and Riedel, S. 2019. Language models as knowledge bases? *arXiv:1909.01066*.

Radford, A.; Narasimhan, K.; Salimans, T.; Sutskever, I.; et al. 2018. Improving language understanding by generative pre-training. *OpenAI*.

Wang, J.; Qu, J.; Liang, Y.; Li, Z.; Liu, A.; Liu, G.; and Zheng, X. 2023. Snowman: A Million-scale Chinese Commonsense Knowledge Graph Distilled from Foundation Model. *arXiv:2306.10241*.