From Discrimination to Generation: 
Knowledge Graph Completion with Generative Transformer

Xin Xie
Zhejiang University
AZFT Joint Lab for Knowledge Engine
Hangzhou, China
xx2020@zju.edu.cn

Ningyu Zhang
Zhejiang University
AZFT Joint Lab for Knowledge Engine
Hangzhou, China
zhangningyu@zju.edu.cn

Zhoubo Li, Shumin Deng
Zhejiang University
AZFT Joint Lab for Knowledge Engine
Hangzhou, China
{mzb,231sm}@zju.edu.cn

Hui Chen
Alibaba Group
Hangzhou, China
weidu.ch@alibaba-inc.com

Feiyu Xiong, Mosha Chen
Alibaba Group
Hangzhou, China
{feiyu.xfy,chenmosha.cms}@alibaba-inc.com

Huajun Chen∗
Zhejiang University & AZFT Joint
Lab for Knowledge Engine
Hangzhou, China
huajunsir@zju.edu.cn

ABSTRACT

Knowledge graph completion aims to address the problem of extending a KG with missing triples. In this paper, we provide an approach GenKGC, which converts knowledge graph completion to sequence-to-sequence generation task with the pre-trained language model. We further introduce relation-guided demonstration and entity-aware hierarchical decoding for better representation learning and fast inference. Experimental results on three datasets show that our approach can obtain better or comparable performance than baselines and achieve faster inference speed compared with previous methods with pre-trained language models. We also release a new large-scale Chinese knowledge graph dataset AliopenKG500 for research purpose.

CCS CONCEPTS

• Computing methodologies → Knowledge representation and reasoning.

KEYWORDS

Knowledge Graph Completion; Generation; Transformer

1 INTRODUCTION

Knowledge Graphs (KGs) treat the knowledge in the real world as fact triples in the form of <subject, predicate, object>, abridged as (s, p, o), where s and o denote entities and p are the relations between entities, which can benefit a wide range of knowledge-intensive tasks. Knowledge graph completion (KGC) aims to complete the knowledge graph by predicting the missing triples. In this paper, we mainly target link prediction task for KG based on the powerful pre-trained language models.

Most previous KG completion methods, such as TransE [2], ComplEx [11], and RotatE [9], are knowledge embedding techniques that embed the entities and relations into a vector space and then obtain the predicted triples by leveraging pre-defined scoring functions to those vectors. Recently, some textual encoding methods (e.g., KG-BERT [14]) have been proposed, which utilize the pre-trained language model to encode triples and output the score for each candidate. Obviously, most previous approaches leverage the discrimination paradigm with a pre-defined scoring function to knowledge embeddings. However, such a discrimination strategy has to costly scoring of all possible triples in inference and suffer from the instability of negative sampling. Moreover, those dense knowledge embedding approaches (e.g., TransE) ignore the fine-grained interactions between entities and relations and have to allocate a large memory footprint for the large-scale real-world knowledge graph. Therefore, it is intuitive to find a new technical solution for knowledge graph completion.

In this paper, we take the first step to model the knowledge graph completion with sequence to sequence generation and propose a novel approach GenKGC. To be specific, we represent entities and relations as input sequences and utilize the pre-trained language model to generate target entities. Following GPT-3’s naive “in-context learning” paradigm, in which the model can learn correct output answers by concatenating the selected samples relevant to the input, we propose relation-guided demonstration by adding triples of the same relation. Moreover, during generation, we propose entity-aware hierarchical decoding to reduce the time complexity of generation. Experimental results on two datasets WN18RR, FB15k-237 and a newly released large-scale Chinese KG dataset AliopenKG500 demonstrate the effectiveness of the proposed approach. The contributions of our work are as follows:
We convert link prediction to sequence to sequence generation and propose GenKGC, which can reduce the inference time while maintaining the performance.

We propose relation-guided demonstration and entity-aware hierarchical decoding, which can better represent entities and relations and reduce the time complexity of generation.

We report results on two datasets and release a new large-scale KG dataset, AliopenKG500, for research purposes.

2 METHOD

2.1 Link Prediction as Seq2Seq Generation

Knowledge graph is defined with entity categories and entity descriptions as a tuple $G = (\mathcal{E}, \mathcal{R}, \mathcal{T}, \mathcal{C}, \mathcal{D})$, where $\mathcal{E}$ represents a set of entities, $\mathcal{R}$ represents relation types, $\mathcal{T}$ refers to a set of triples, $\mathcal{C}$ refers to the entity categories and $\mathcal{D}$ refers to the entity descriptions. For each triple $t \in \mathcal{T}$, there is the form $(e_i, r_j, e_k)$ where $e_i, e_k \in \mathcal{E}$ is the head and tail entity respectively. For each entity $e_i \in \mathcal{E}$, there exists category $c_i$ to define the $e_i$ and a text $d_i$ to describe $e_i$. To complete missing triples in KGs, link prediction aims at predicting the tail entity given the head entity and the query relation, denoted by $(e_i, r, \_)$.

In this paper, we utilize the standard encoder-decoder architecture for sequence-to-sequence generation. Note that we regard each entity and relation as the sequence of tokens. Concretely, we follow [14] to use plain text to represent the entities and relations instead of unique embedding to bridge the gap between the triples in the knowledge graph and pre-trained language models. To be specific, given a triple with tail entity missing $(e_i, r_j, \_)$, we obtain the description $d_e$, $d_r$ of $e_i$ and $r_j$ respectively, and concatenate them together. For example, as shown in Figure 1. "Michael Chabon studied at" (head entity, relation) is the major part of the input sequence, and "UC, Irvine" (target entity) is part of the output sequence. Thus, we have the input sequence of $<e_i, r>$ pair and generate the output sequence of $e_k$. We leverage BART for model training and inference.

Table 1: Inference and training efficiency comparison. $|d|$ is the length of the entity description. $|\mathcal{E}|$ is the numbers of all unique entities in the KG. $k$ is the negative samples for KG-BERT and the beam size for our GenKGC. The time under RTX 3090 is used to estimate the speed of training and inference given a single (entity,relation) pair on AliopenKG500.

| Method     | Complexity     | Time under RTX 3090 |
|------------|----------------|---------------------|
| Training   | $O(|d|^2 \times |\mathcal{E}|)$ | 72ms               |
| GenKGC     | $O(|d|^2)$     | 2.35ms              |
| Inference  | $O(|d|^2 \times |\mathcal{E}|)$ | 10100s             |
| GenKGC     | $O(|d|^2 \times |d|^k)$ | 0.96s              |

2.2 Relation-guided Demonstration

Inspired by prompt-tuning [18], we propose relation-guided demonstration for the encoder. Note that there exist long-tailed distributions in the KGs, for example, there exist only 37 instances of the relation film/type_of_appearance in the FB15k-237 dataset. Previous studies [18] illustrate that concatenating randomly sampled instances as demonstrations can yield better few-shot performance. Thus, we construct relation-guided demonstration examples $\{t_{in, \text{train}}\}$. To be specific, we sample those demonstrations with the guidance of relation $r_j$, which consists of several triples with the same relation of input from training set. Formally, we have the final input sequence as:

$x = \langle \text{bos} \rangle \text{ demonstration}(r_j) \langle \text{sep} \rangle d_e, d_r \langle \text{sep} \rangle$

2.3 Entity-aware Hierarchical Decoding

In the vanilla decoding setting, we have to enumerate all entities in the $\mathcal{E}$ and then sort them by the score function. However, this process can be time-consuming, as shown in Table 1 when $\mathcal{E}$ is very large, it is costly scoring of all possible triples. In our approach, we follow [3] to use the Beam Search to obtain top-$k$ entities in the $\mathcal{E}$ ($k$ is the hyperparameter of beam size). Intuitively, there is no need for negative sampling as we directly optimize by predicting the correct...
When the candidate entities set is huge, it is time-consuming for
entity in decoding. To be more specific, given with a triple with
tail or head entity missing \((e_i, r_j, ?)\), GenKGC rank each
\(e \in E\) by computing a score with an autoregressive formulation:

\[
p_\theta(y \mid x) = \prod_{i=1}^{|c|} p_\theta(z_i \mid z_{<i}, x) \prod_{i=|c|+1}^N p_\theta(y_i \mid y_{<i}, x), \tag{1}
\]

where \(z\) is the set of \(|c|\) tokens in the category \(c\), \(y\) is the set of \(N\) tokens in the textual representation of \(e\).

Since KG contains rich semantic information such as entity types,
it is intuitive to constrain the decoding for fast inference. We sample
the type category with the lowest frequency of occurrence in the
training set to constrain the entity decoding since it is challenging
to discriminate those low-frequent entities. Then, we add special
tokens as types in the vocabulary of pre-trained language model
to constrain the decoding. To ensure that the generated entities
are inside the entity candidate set, we construct a prefix tree (trie
tree) to decode our entity name as shown in Figure 1. Similar to
the ordinary sequence-to-sequence model, we optimize GenKGC
using a standard seq2seq objective function:

\[
L = -\log p_\theta(y \mid x) \tag{2}
\]

### 3 EXPERIMENT

**Datasets.** We evaluate our method on FB15k-237 [10], WN18RR [5], which are widely used in the link prediction literature, and a new real-world large-scale Chinese KG dataset AliOpenKG500\(^2\). In FB15k-237, descriptions of entities are obtained from the intro-
duction section of the Wikipedia page of each entity. In WN18RR,
each entity corresponds to a word sense, and description is the
word definition. In AliOpenKG500, the descriptions of entities and
relations come from the e-commerce description page. More details
about datasets are listed in Table 3.

\(^2\)AliOpenKG500 is a subset of real-world open business knowledge graph from https://kg.alibaba.com/.

### Table 2: Experiment results on WN18RR, FB15k-237 and AliopenKG500. ∗Resulting numbers are reported by [8], we reproduce
the model result on AliopenKG500 and take other results from the original papers.

| Method       | WN18RR | | | | FB15k-237 | | | | AliopenKG500 | | |
|--------------|--------|---|---|---|--------|---|---|---|----------------|---|---|---|---|
|              | Hits@1 | Hits@3 | Hits@10 | | Hits@1 | Hits@3 | Hits@10 | | Hits@1 | Hits@3 | Hits@10 | |
| TransE [2]   | 0.043  | 0.441 | 0.532 | 0.198 | 0.376 | 0.441 | 0.207 | 0.340 | 0.513 | |
| DistMult [13] | 0.412  | 0.470 | 0.504 | 0.199 | 0.301 | 0.446 | 0.049 | 0.088 | 0.216 | |
| ComplEx [11] | 0.409  | 0.469 | 0.530 | 0.194 | 0.297 | 0.450 | 0.053 | 0.120 | 0.266 | |
| RotatE [9]   | 0.428  | 0.492 | 0.571 | 0.241 | 0.375 | 0.533 | -    | -    | -    | |
| TuckER [1]   | 0.443  | 0.482 | 0.526 | 0.226 | 0.394 | 0.544 | -    | -    | -    | |
| ATTH [4]     | 0.443  | 0.499 | 0.486 | 0.252 | 0.384 | 0.549 | -    | -    | -    | |

### Table 3: Summary statistics of benchmark datasets.

| Dataset       | # Ent | # Rel | # Train | # Dev | # Test |
|---------------|-------|-------|---------|-------|--------|
| WN18RR        | 40,943| 11    | 86,835  | 3,034 | 3,134  |
| FB15k-237     | 14,541| 237   | 272,115 | 17,535| 20,466 |
| AliOpenKG500  | 269,658| 500  | 1,242,550| 5,000 | 5,000 |

**Setting.** We adopt BART-base as our backbone for comparison
with other BERT-based KGC methods like KG-BERT. Following
StAR [12], we choose the two kinds of KGE methods as our baseline
models, which can be classified as graph embedding approach and
textual encoding approach. Grid search is used over the three
datasets and the results are reported on the test set with hyperparameters
of the best performance determined by the dev set.

**Metrics.** We evaluate the test set of triples \(T^*\) disjoint from
the set of training triples \(T\). For inference on a test triple \((e_i, r_j, r_k)\),
we make sure predict the entity in the candidate set \(E\). We use the
metrics of hits@1, hits@3 and hits@10. We use the metrics of hits@1, hits@3 and hits@10.

**Comparison with Discrimination Method.** From Table 2, we notice
that GenKGC achieves better performance than KG-BERT [14]
across all datasets and maintains high speed during inference. The
translation-based method like TransE, which treats entities and
relations as dense vectors in the same space, will face the memory
explosion problem. For the memory cost, TransE has to consume 260M
parameters to store the entities and relations for AliopenKG500
with more than 260k entities, while pre-trained models (BERT or
BART) only utilize 110M parameters, which demonstrates the memory
efficiency of our approach. Note that this problem will be more
severe when the entities become more numerous because the space
complexity is \(O(n)\). For the inference speed, KG-BERT encodes the
relational triples with the pre-trained language model and ranks
all candidate triples with correct and wrong entities for inference.
When the candidate entities set is huge, it is time-consuming for
inference; for example, as shown in Table 1, KG-BERT takes about
100100s to the scoring of all possible triples given a single (entity,
Table 4: We list a query and first five entities with their probability predicted by GenKGC w/o entity-aware decoding, and its reranking with GenKGC.

| Rank | GenKGC w/o hierarchical decoding | Probability |
|------|----------------------------------|-------------|
| 1    | University of California         |             |
| 2    | University of California, Irvine |             |
| 3    | University of California, San Francisco |   |
| 4    | University of California, Davis  |             |
| 5    | University of California, Santa Cruz |        |

| Rank | GenKGC | Probability |
|------|--------|-------------|
| 1    | University of California, Irvine |   |
| 2    | University of California, San Francisco | |
| 3    | University of California, Davis |       |
| 4    | University of California, Santa Cruz | |
| 5    | University of Calgary             |       |

relation) pair. While our method only needs to generate the top-k entities with entity-aware hierarchical decoding, which reduces lots of computing resources.

3.1 Case Study

For different decoding strategies we conduct case study to analyze the result. For GenKGC w/o hierarchical decoding, we utilize the normal beam search to decode the text name of the missing entity. From Table 4, we observe that GenKGC obtain better entity generation results while in normal beam search, the model may stop early at correct but not precise enough answer. We argue that this is caused by the bias of the pre-trained language model (e.g., common token bias) since high-frequent tokens will lead the pre-trained language model to be biased toward certain answers. Our entity-aware hierarchical decoding can constrain the decoding process and mitigate the bias effect caused by pre-trained language models.

4 RELATED WORK

Knowledge Graph Embedding Models. There are lots of methods of KGC are based on knowledge graph embeddings (KGE), which generally leverage an embedding vector in the continuous embedding space to represent the entity and the relation in KG [17]. One kind of KGE methods is translation-based, such as TransE [2], ComE [20], T2E [9], which consider relations as the mapping function between entities by using distance-based scoring functions. The other kind of KGE method is semantic matching models, where they get the semantic similarity by using the multi-linear or bilinear product.

Pre-trained Language Models for KGC. Recently, since pre-trained language models, such as BERT [6], have shown significant improvement on several natural language processing tasks, several works use the transformer-based models to tackle the KG problems [15, 16, 19]. KG-BERT [14] first propose to use BERT for KGC by seeing a triple as a sequence and converts KGC into a sequence classification problem with the binary cross-entropy object. [7] proposes to use a transformer encoder-decoder model that takes plain text as input and output a structured triple of the information hide in it.

5 CONCLUSION

In this paper, we propose GenKGC, which can reach comparable results while reducing inference and training cost in link prediction with pre-trained models. Experimental results on three benchmark datasets demonstrate the effectiveness of our approach, especially in inference time. The success of GenKGC suggests that link prediction as sequence-to-sequence generation can be helpful because entities can be inferred directly rather than costly scoring of all possible triples. Future work should explore how to obtain the suitable categories for each entity in decoding.

REFERENCES

[1] Ivana Balazevic, Carl Allen, and Timothy M. Hospedales. 2019. TuckER: Tensor Factorization for Knowledge Graph Completion. In Proc. of EMNLP.
[2] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oak-sana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In Proc. of NeurIPS.
[3] Nicola De Cao, Gautier Izacard, Sebastian Riedel, and Fabio Petroni. 2021. Autoregressive Entity Retrieval. In Proc. of ICLR.
[4] Ines Chami, Adva Wolf, Da-Cheng Juan, Frederic Sala, Sujith Ravi, and Christopher Ré. 2020. Low-Dimensional Hyperbolic Knowledge Graph Embeddings. In Proc. of NeurIPS.
[5] Tim Dettmers, Pasquesi Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Convolutional 2D Knowledge Graph Embeddings. In Proc. of AAAI.
[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Proc. of NAACL.
[7] Martin Josifoski, Nicola De Cao, Maxime Peyrard, and Robert West. 2021. GenIE: Generative Information Extraction. CoRR (2021).
[8] Deepak Nathani, Jain Chauhan, Charu Sharma, and Manohar Kaul. 2019. Learning Attention-based Embeddings for Relation Prediction in Knowledge Graphs. In Proc. of ACL.
[9] Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019. RotatE: Knowledge Graph Embedding by Rotational Relation in Complex Space. In ICLR.
[10] Kristina Toutanova, Danqi Chen, Patrick Pantel, Houfeng Pou, Pallavi Choudhury, and Michael Gamon. 2015. Representing Text for Joint Embedding of Text and Knowledge Bases. In Proc. of EMNLP.
[11] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Eric Gaussier, and Guillaume Bouchard. 2016. Complex Embeddings for Simple Link Prediction. In Proc. of ICLR.
[12] Bo Wang, Tao Shen, Guodong Long, Tianyi Zhou, Ying Wang, and Yi Chang. 2021. Structure-Augmented Text Representation Learning for Efficient Knowledge Graph Completion. In Proc. of WWW.
[13] 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 Proc. of ICLR.
[14] Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. KG-BERT: BERT for Knowledge Graph Completion. CoRR (2019).
[15] Hongzhe Ye, Ningyu Zhang, Shumin Deng, Xiang Chen, Hui Chen, Feiyu Xiong, Xi Chen, and Huajun Chen. 2022. OntoProtein: Protein Embedding Entities and Relations for Learning and Inference in Knowledge Bases. In Proc. of ICLR.
[16] Ningyu Zhang, Zhou Li, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding Entities and Relations for Learning and Inference in Knowledge Bases. In Proc. of ICLR.
[17] Ningyu Zhang, Zhou Li, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding Entities and Relations for Learning and Inference in Knowledge Bases. In Proc. of ICLR.
[18] Ningyu Zhang, Zhou Li, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding Entities and Relations for Learning and Inference in Knowledge Bases. In Proc. of ICLR.
[19] Ningyu Zhang, Zhou Li, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding Entities and Relations for Learning and Inference in Knowledge Bases. In Proc. of ICLR.