A Knowledge-Injected Curriculum Pretraining Framework for Question Answering

Xin Lin
School of Computer Science and Technology, University of Science and Technology of China & State Key Laboratory of Cognitive Intelligence
Hefei, China
linx@mail.ustc.edu.cn

Tianhuang Su
Guangdong OPPO Mobile Telecommunications Corp., Ltd
Shenzhen, China
sutianhuang@oppo.com

Zhenya Huang*
School of Computer Science and Technology, University of Science and Technology of China & State Key Laboratory of Cognitive Intelligence
Hefei, China
huangzhy@ustc.edu.cn

Shangzi Xue
School of Computer Science and Technology, University of Science and Technology of China & State Key Laboratory of Cognitive Intelligence
Hefei, China
xueshangzi@mail.ustc.edu.cn

Haifeng Liu
University of Science and Technology of China
Hefei, China
bladehliu@qq.com

Enhong Chen
Anhui Province Key Laboratory of Big Data Analysis and Application, University of Science and Technology of China & State Key Laboratory of Cognitive Intelligence
Hefei, China
cheneh@ustc.edu.cn

ABSTRACT

Knowledge-based question answering (KBQA) is a key task in natural language processing research, and also an approach to access the web data and knowledge, which requires exploiting knowledge graphs (KGs) for reasoning. In the literature, one promising solution for KBQA is to incorporate the pretrained language model (LM) with KGs by generating KG-centered pretraining corpus, which has shown its superiority. However, these methods often depend on specific techniques and resources to work, which may not always be available and restrict its application. Moreover, existing methods focus more on improving language understanding with KGs, while neglect the more important human-like complex reasoning. To this end, in this paper, we propose a general **K**nowledge-**I**njected **C**urriculum **P**retraining framework (KICP) to achieve comprehensive KG learning and exploitation for KBQA tasks, which is composed of knowledge injection (KI), knowledge adaptation (KA) and curriculum reasoning (CR). Specifically, the KI module first injects knowledge into the LM by generating KG-centered pretraining corpus, and generalizes the process into three key steps that could work with different implementations for flexible application. Next, the KA module learns knowledge from the generated corpus with LM equipped with an adapter as well as keeps its original natural language understanding ability to reduce the negative impacts of the difference between the generated and natural corpus. Last, to enable the LM with complex reasoning, the CR module follows human reasoning patterns to construct three corpora with increasing difficulties of reasoning, and further trains the LM from easy to hard in a curriculum manner to promote model learning. We provide an implementation of the general framework, and evaluate the proposed KICP on four real-word datasets. The results demonstrate that our framework can achieve higher performances, and have good generalization ability to other QA tasks.

CCS CONCEPTS

• **Computing methodologies** → **Knowledge representation and reasoning**.

KEYWORDS

Question answering, Knowledge-injected pretraining, Curriculum learning

ACM Reference Format:
Xin Lin, Tianhuang Su, Zhenya Huang, Shangzi Xue, Haifeng Liu, and Enhong Chen. 2024. A Knowledge-Injected Curriculum Pretraining Framework for Question Answering. In *Proceedings of the ACM Web Conference 2024 (WWW '24), May 13–17, 2024, Singapore, Singapore.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3589334.3645406

1 INTRODUCTION

Knowledge-based question answering (KBQA) is a key task in natural language processing (NLP) and data mining research [33], which could act as an approach to access and process web data and knowledge, and lead to useful applications such as smart voice assistant and search engine especially with the large language models (LLMs) [30]. As shown in Figure 1, KBQA aims to answer questions in natural language based on background knowledge, which is often formatted as knowledge graphs (KGs) [19, 45, 49]. Therefore, KBQA requires abilities of both natural language understanding

*Corresponding Author.

WWW '24, May 13–17, 2024, Singapore, Singapore

ACM ISBN 979-8-4007-0171-9/24/05…$15.00
https://doi.org/10.1145/3589334.3645406

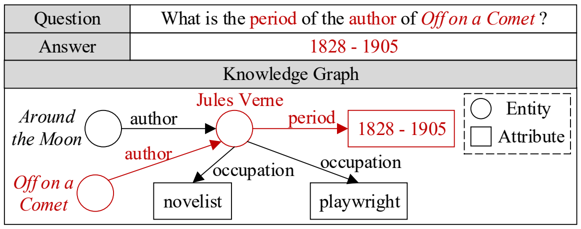


Figure 1: A toy example of KBQA, which requires complex reasoning marked in red.

(NLU) and knowledge reasoning, making it a challenging task in related fields.

In the literature, researchers have proposed many solutions for KBQA [28, 33, 49] based on deep learning due to their remarkable results on other NLP tasks [12, 15, 18], among which the pretrained language models (LMs) have become the most promising for its strong NLU ability [4, 6, 29, 30]. Unfortunately, LMs including the LLMs work not so well in knowledge application [21, 23], which hinders its application in KBQA. Therefore, researchers have tried great efforts to enhance the LMs with KGs (inputting knowledge facts into LMs, or pretraining LMs with knowledge-based tasks [21, 31, 34, 36, 37, 46, 47, 50, 53]), which has greatly improved LMs in knowledge-related tasks. However, these methods often learn KGs as supplementary to additional pretraining corpus [21, 50], which can not cover the whole KG and may overlook some knowledge useful in certain tasks, and thus leads to incomplete knowledge learning. Towards this point, a straightforward solution is to generate the pretraining corpus based on the KGs. Although many methods have been developed along this line [2, 3, 20, 48], they usually depend on specific techniques or resources for effective corpus generation (e.g., requiring pretrained generative model to generate sentences, or generating sentences in a fixed format), which may be unavailable in practice and thus restricts its application. Therefore, in this paper we hope to design a general framework to generate KG-centered corpus for comprehensive knowledge pretraining of LMs, which is not limited to specific techniques and could work with different detailed implementations for flexible application.

However, along this line there exist several nontrivial technical challenges. First, there are many solutions to generate sentences based on given KGs for different demands (e.g., pretrained generative LMs [2], fixed sentence templates [20]). Moreover, although most KGs store the knowledge triples with entity IDs, some high-quality KGs also contain additional attribute information, which is stored in various forms (e.g., texts, numbers and dates) and requires different processing. How to unify and generalize these various techniques and data forms remains much open. Second, the generated sentences differ from natural ones and may even seem distorted, which may mislead the LM and hurt natural language understanding ability of the LM in pretraining [2, 20]. Existing methods address this problem with specific techniques in accordance with their generation methods (e.g., generating sentences more similar to natural ones with complex generative LMs [2], or adopting specially designed sentence templates to reduce the negative impacts [20]), but how to overcome this shortcoming for an arbitrary generation method in the general framework is a nontrivial problem. Last, existing methods enhancing LMs with KGs focus more on improving language understanding with related knowledge such as K-BERT [21] and ERNIE [50], while seldom have considered the human-like complex reasoning ability. Humans can perform reasoning over multiple knowledge facts following specific patterns, which is also widely required in KBQA tasks. For example, in Figure 1, to reach the answer, the LM first needs to find that the author of *Off on a Comet* is Jules Verne, and then the period of Jules Verne is 1828-1905. How to enable the LMs with such complex reasoning is a challenging problem.

To this end, in this paper, we propose a general **K**nowledge-**I**njected **C**urriculum **P**retraining framework (KICP) to achieve comprehensive KG learning and exploitation for KBQA, which is composed of knowledge injection (KI), knowledge adaptation (KA) and curriculum reasoning (CR). Specifically, the KI module converts KG triples into sentences to construct pretraining corpus for complete knowledge learning, and generalizes the process into three key steps, i.e., text characterization, sentence construction and masking, which can be implemented with different detailed techniques and various data forms for flexible application. Next, to reduce the negative impacts brought by the difference between generated and natural corpus on LM pretraining, the KA module fixes the original LM to keep its NLU ability, and learns knowledge from the generated corpus with a trainable adapter working with the LM. Last, to pretrain the LM with complex reasoning ability, the CR module follows common reasoning patterns of humans and constructs corpora requiring complex knowledge reasoning. Furthermore, the CR module arranges the complex corpora into three lessons with increasing difficulties, and trains the LM from easy to hard following the curriculum learning manner to reduce pretraining difficulty. Finally, we provide an implementation of the general framework, and conduct extensive experiments on four real-word datasets to evaluate KICP. The results demonstrate that our framework can achieve higher performances, and generalize to other QA tasks well.

2 RELATED WORK

Knowledge-Based Question Answering. In the literature, studies on KBQA can be roughly divided into the knowledge-enhanced LM (introduced later), and the KG-based reasoning including path-based [27], embedding-based [11, 33] and graph-based methods [10, 14, 28, 44, 45, 49]. Path-based methods map the question into entities and relations for reasoning on the KG [27]. Embedding-based methods such as EmbedKGQA [33] represent the question and KG in the same latent space, and infer the answer with simple vector computation. Graph-based methods [10, 14, 28, 44, 45, 49] sample a sub-graph from the KG, and perform reasoning on the sub-graph with neural networks. Graph-based methods are widely applied in complex reasoning for the good trade-off between interpretability and performance, but the insufficient knowledge modeling within the sub-graph may lead to limited robustness. Besides, the large language models (LLMs) have become a promising method in KBQA

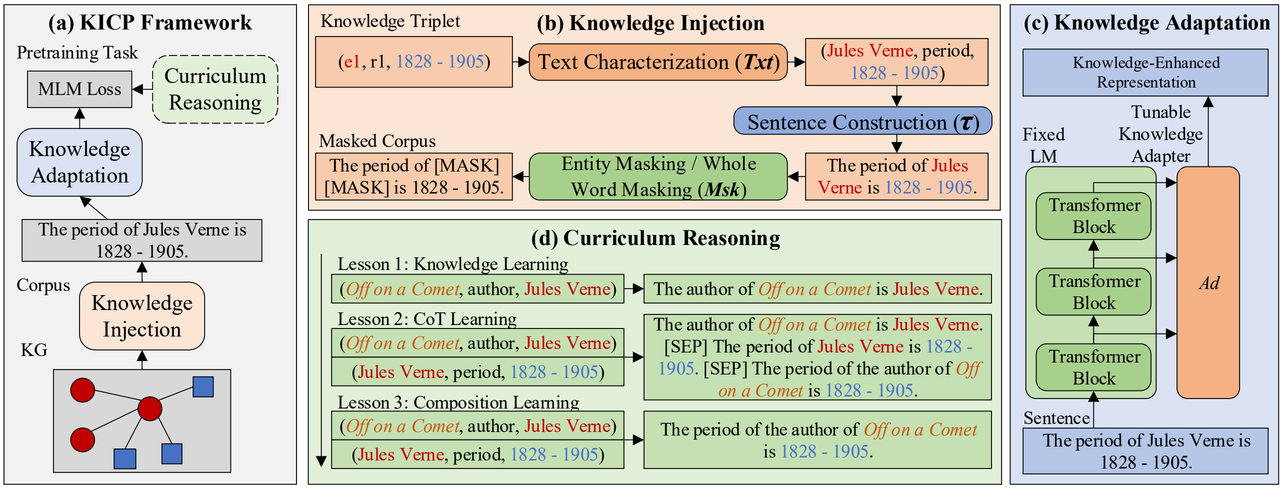


Figure 2: The architecture of the proposed KICP framework. (a) The overview of KICP. (b) The knowledge injection module (KI) converts KG triples into sentences. (c) The knowledge adaptation module (KA) works with the LM to keep NLU ability and learn knowledge. (d) The curriculum reasoning module (CR) constructs easy-to-hard reasoning-required pretraining corpora.

tasks recently [1, 8]. Researchers have proposed several advanced techniques to improve its knowledge reasoning ability, including the chain-of-thought prompt [39], question decomposition [52], and retrieval augmented generation [43].

Knowledge-Enhanced Language Model. As the pretrained LMs have shown its weakness on knowledge-based tasks [21, 23], researchers have tried many efforts to enhance LMs with knowledge from KGs, including the explicit methods [21, 26, 31, 50] and implicit methods [9, 20, 34, 36, 37, 42]. Explicit methods feed knowledge facts or embeddings into LM as additional inputs to exploit related knowledge. For example, K-BERT [21] injected the knowledge triples into the sentences as inputs to the LM. Zhang *et al.* [50] developed an aggregator network to incorporate KG entity embeddings into LMs. Implicit methods design special pretraining tasks to learn knowledge from KGs and corpus with LM. Sun *et al.* [34] introduced an entity masking strategy for pretraining, and Wang *et al.* [37] trained LM as knowledge embedding model with entity descriptions. To better exploit KG triples, Liu *et al.* [20] generated multilingual synthetic pretraining corpus with KG triples and Agarwal *et al.* [2] employed the generative LM to synthesize more natural corpus. In summary, explicit methods exploit the knowledge more directly but require additional knowledge annotations as inputs, while implicit methods can be easily applied in downstream tasks, but require heavy pretraining.

Our work differs from previous methods as follows. First, existing methods converting the KG into corpus are often limited to specific techniques and resources, while our method is a general framework which can work with different detailed implementations for different circumstances. Second, existing methods focus more on improving language understanding with related knowledge, while our method further enables the LM with complex reasoning ability with specially designed pretraining task.

3 KICP: KNOWLEDGE-INJECTED CURRICULUM PRETRAINING

3.1 Problem Definition

Knowledge-based question answering (KBQA) is composed of the knowledge graph $\mathcal{KG}$ and the question-answer pair (Q, Y). We suppose that the KG contains knowledge triples about the relation between two entities and the attribute of each entity, where the attribute values are in diverse forms that can be converted into texts (texts are defined as V^+ on vocabulary V). Therefore, the KG can be defined as $\mathcal{KG} = (\mathbb{E}, \mathbb{R}, \sum)$, where $\mathbb{E}$ is the entity set, $\mathbb{R}$ is the relation and attribute set, and $\sum$ means the knowledge triples. Each triple $(h, r, t) \in \sum$ $(h, t \in \mathbb{E}, r \in \mathbb{R})$ means that the entity h and t have the relation r (e.g., "Jules Verne" is the "author" of "*Off on a Comet*" in Figure 1), and $(h, r, t) \in \sum$ $(h \in \mathbb{E}, r \in \mathbb{R}, t \in V^+)$ means the attribute r of entity h is t, where t is the attribute value in text (e.g., the "period" of "Jules Verne" is "1828-1905") Besides, each entity $e \in \mathbb{E}$ is assigned with several names $N_e = \{n_{e1}, n_{e2}, \ldots, n_{ek}\}$ (each name $n_{ei} \in V^+$). $\mathbb{R}$ is assigned with names similarly. In the question-answer pair (Q, Y), $Q = \{q_1, q_2, \ldots, q_n\} \in V^+$ $(q_i \in V)$ is the question in natural language, and Y is the answer to Q inferred under $\mathcal{KG}$, whose form depends on the task (e.g., KBQA often selects an entity or attribute value from $\mathcal{KG}$, and generative QA generates formal language from certain vocabulary such as natural text or mathematical expression [16, 17]).

Given the knowledge graph $\mathcal{KG}$ and question-answer pair (Q, Y), the goal of KBQA is to train a model M :$(\mathcal{KG}, Q) \rightarrow Y$ to predict the answer Y of question Q under $\mathcal{KG}$. In this paper, we first pretrain a language model $\mathcal{LM}$ with $\mathcal{KG}$, and then use it in M to predict the answer Y to Q. We expect that $\mathcal{LM}$ could learn knowledge from $\mathcal{KG}$ comprehensively and well handle complex reasoning.

3.2 Method

We propose a general **K**nowledge-**I**njected **C**urriculum **P**retraining framework (KICP) to pretrain $\mathcal{LM}$ for comprehensive knowledge learning and complex reasoning, which is not limited to specific techniques and could easily work with different implementations for flexible applications. As shown in Figure 2 (a), KICP is composed of three key components, i.e., *knowledge injection* (KI), *knowledge adaptation* (KA) and *curriculum reasoning* (CR). Specifically, KI injects knowledge from the KG into the LM completely by converting the KG triples to sentences to construct the pretraining corpus, and generalize the various generation techniques into three key steps. To reduce the negative impacts brought by the gap between generated and natural corpus, KA fixes the original LM to keep its NLU ability, and equips the framework with a trainable knowledge adapter to learn knowledge from the generated corpus. To pretrain the LM with complex reasoning ability, CR follows common patterns of human reasoning and constructs several reasoning-required corpora with different difficulties, and trains the LM from easy to hard in a curriculum manner to promote model learning.

3.2.1 Knowledge Injection. To overcome the insufficient knowledge learning brought by using the KG as supplementary to external corpus, we directly convert the KG triples into sentences as pretraining corpus to inject knowledge into the LM. Moreover, there exist several effective sentence generation techniques for different requirements in the literature [2, 20], and the KGs contain multiple forms of data that requires different processing (e.g., IDs, texts, numbers and dates). Therefore, to generalize these detailed techniques to a general framework that is not limited to specific techniques for flexible application in various circumstances, as shown in Figure 2 (b), we abstract the sentence generation process into three key steps, i.e., text characterization, sentence construction and masking.

Text Characterization. Given one triple $k = (h, r, t) \in \sum$ sampled from $\mathcal{KG}$, KI first characterizes all fields of the triple as texts (*Txt*), which serve as the backbone elements of the sentence to generate. For the entities and relations stored in IDs, We map the meaningless ID (e.g., e1) to a meaningful name (Jules Verne), which is dynamically sampled from the associated name set in each iteration to increase corpus diversity. More sampling strategies can also be applied here for other demands [20]. For the various forms of attribute values (e.g, numbers, dates and texts), we use their textual descriptions as they can always be expressed with texts despite the original forms. In this way, we can unify the diverse processing of the entities, relations and attribute values.

Sentence Construction. After getting the textual elements, KI applies a sentence construction strategy τ to assemble these elements into a complete sentence, including reordering and transforming the elements and adding auxiliary words. The strategy τ can be implemented with different existing techniques, such as sentence templates, grammar-based rules, and the generative LMs [2, 20].

Masking. The last step is to mask the generated sentence for masked language model (MLM) pretraining. To force knowledge learning and match the differences between entities and attribute values, we prefer paying more weights to the knowledge elements in the sentence (those converted from the triple), and applying different masking strategies *Msk* to entities and attribute values. For example, we apply the entity masking [34] on entities which masks the whole entity name to force learning relation knowledge instead of memorizing the entity name, and whole word masking (WWM) [5] on attribute values since the values may contain too much information (e.g., biography) and are too hard to recover if all masked. WWM also works similarly to entity masking on short values (e.g., numbers) by masking as a whole word. More masking techniques can be used here as *Msk*.

Overall, the sentence generation process is formulated as follows:

$$KI(k) = Msk(\tau(Txt(h), Txt(r), Txt(t))), \quad k = (h, r, t) \in \sum . \tag{1}$$

The knowledge-injected corpus is composed of the sentences $KI(k)$, which are dynamically generated from triples sampled from the KG in pretraining. In this way, KI converts the whole KG into the corpus, and thus implicitly stores all information from the KG in the corpus such as the structural infromation. Compared with existing methods rewriting KG as corpus, KI does not depend on specific techniques or resources, and thus could work with different implementations for various application demands.

3.2.2 Knowledge Adaptation. Obviously the corpus generated by KI differs from natural ones as the sentences may not strictly follow the grammar (especially for some simple τ), and the diversity of the corpus is limited. Pretraining the LM on the corpus may hurt NLU ability and work badly on natural texts. Furthermore, as the sentence generation technique in the proposed general framework is arbitrary, we can not use methods associated with specific generation techniques to address the problem as existing studies [2, 20]. Therefore, in knowledge adaptation (KA), we turn to keeping the NLU ability of LM during knowledge pretraining.

As demonstrated by Figure 2 (c), following the adapter paradigm in LM tuning [7, 36], we fix the LM parameters and add a trainable knowledge adapter module *Ad* above the original LM *LM*. *Ad* uses the semantic outputs of *LM* as inputs, and outputs the knowledge-enhanced representations. Moreover, to deeply improve the fusion of the semantics and knowledge, the semantic outputs of all layers in the LM are used. The computation of KA is formulated as follows:

$$KA(x) = Ad(LM(x)), \tag{2}$$

where x is the input sentence. *Ad* can be implemented with any neural networks, which is expected to have a proper size to contain enough space for knowledge learning and avoid greatly increasing computation complexity as well.

In pretraining, the parameters of *Ad* is trained to learn knowledge from the constructed corpus, while the original LM is fixed. As the original LM is not affected by *Ad*, the NLU ability is retained as much as possible to reduce the negative impacts of the gap between generated and natural corpus.

3.2.3 Curriculum Reasoning. With KI and KA, KICP can effectively inject the KG into LM, but still lacks complex reasoning ability over multiple knowledge facts as required in real-world KBQA tasks. To enable the LM with such ability, the curriculum reasoning module (CR) pretrains LM on corpora requiring complex reasoning as shown in Figure 2 (d).

It is hard to collect enough reasoning-required corpus for all KGs, so we also build the corpus based on the KG. Humans often perform complex reasoning following specific patterns (e.g., multi-top reasoning), which put restrictions on the participating triples (e.g., the chain-like triples). Therefore, we build the corpus following these patterns (e.g., "The period of the author of *Off on a Comet* is 1828-1905"). We first sample several triples $\{k_1, \ldots, k_n\}$ matching the restrictions from KG, such as the chain-like triples {(*Off on a Comet*, author, Jules Verne), (Jules Verne, period, 1828-1905)} for multi-hop reasoning, and then convert them into a complex composition with a pipeline *Comp* similar to KI as follows:

$$\begin{aligned} Comp(k_1, \ldots, k_n) =& Msk'(\tau'(Txt(h_1), Txt(r_1), Txt(t_1), \\ & \ldots, Txt(t_n))), \quad k_i = (h_i, r_i, t_i) \in \sum, \end{aligned} \tag{3}$$

where τ' and Msk' are sentence construction and masking in *Comp*. In this way, the complex corpus matches human reasoning, and explicitly exploits the structural information from the KG as well. Much more reasoning patterns can be supported by the CR module.

The complex composition often discards some information to infer from knowledge, so it is hard to pretrain LM directly (e.g., in previous example "Jules Verne" is discarded, which makes it hard to understand without related knowledge). Therefore, as shown in Figure 2 (d), we split the pretraining into three lessons with generated corpora from easy to hard following curriculum learning [51] to promote model learning.

Lesson 1: Knowledge Learning. We start by pretraining LM on single triples from the KG. We build this corpus with KI based on one triple k for each sentence, and pretrain the LM (i.e., KA) on the MLM task to memorize the knowledge facts as follows:

$$\min_{\theta_{Ad},\ \theta_{MLM}} L_1(k) = MLM(KA(KI(k))), \tag{4}$$

where θ_{Ad} and θ_{MLM} means trainable parameters for knowledge adapter *Ad* in *KA* and MLM head.

Lesson 2: CoT Learning. Having learned basic knowledge facts from KG, next we teach the LM how to conduct complex reasoning with related knowledge facts. Inspired by chain-of-thought (CoT) [25, 40], we assemble each sentence with complex composition by *Comp* for certain reasoning pattern and all related knowledge by *KI* as reasoning steps base on triples $\{k_1, \ldots, k_n\}$. To avoid information leakage, we mask the same element (e.g., entity) in both the final composition and reasoning steps, and pretrain the LM on the MLM task as follows:

$$\begin{aligned} \min_{\theta_{Ad},\ \theta_{MLM}} L_2(k_1, \ldots, k_n) =& MLM(KA([KI(k_1), \ldots, \\ & KI(k_n), Comp(k_1, \ldots, k_n)])), \end{aligned} \tag{5}$$

where [,] means text concatenation, and $\{k_1, \ldots, k_n\}$ matches the reasoning pattern for *Comp*.

Lesson 3: Composition Learning. In the hardest lesson, we pretrain the LM to reason with memorized knowledge as real-world QA tasks, where we only provide the final compositions without related reasoning steps. Therefore, We construct the corpus with the complex compositions by *Comp*, and pretrain the LM on the MLM task as follows:

$$\min_{\theta_{Ad},\ \theta_{MLM}} L_3(k_1, \ldots, k_n) = MLM(KA(Comp(k_1, \ldots, k_n))). \tag{6}$$

The corpora are dynamically generated with randomly sampled triples in pretraining. We demonstrate some samples of corpora in three lessons in Appendix D. Through the three pretraining lessons, we explicitly enable the LM with human-like complex reasoning ability required in KBQA tasks, and reduce the pretraining difficulty with the curriculum learning.

3.2.4 QA Fine-Tuning. After pretrained on the KG, the LM can be easily applied in different downstream QA tasks without additional annotations or external knowledge inputs. Specifically, the LM (i.e., *KA*) reads the question Q as input, and outputs the knowledge-enhanced vector, which is fed to a task-dependent prediction head *Pred* to generate the answer Y. The whole system (*LM* and *Ad* in *KA* and *Pred*) can be fine-tuned on different QA tasks subject to the task-dependent objective function $\mathcal{L}$ as follows:

$$\min_{\theta_{LM},\ \theta_{Ad},\ \theta_{Pred}} L_{QA}(Q, Y) = \mathcal{L}(Pred(KA(Q)), Y), \tag{7}$$

where θ_{LM}, θ_{Ad} and θ_{Pred} are parameters of these modules.

3.3 Implementation

In this section, we provide an implementation of the general KICP framework. In KI, we implement text characterization and masking as mentioned in section 3.2.1, and realize τ by simply concatenating all fields, which works well on our datasets.

In KA, we implement the knowledge adapter *Ad* as BERT with the same number of layers and halved vector dimension. In each layer of *Ad*, the input (semantic vector from corresponding layer of *LM*) is first projected with a linear model to the latent space of hidden vector from last layer, and then added with the hidden vector to feed to the BERT layer. The final vectors of *Ad* and *LM* are merged with a linear layer as the output.

In CR, we implement *Comp* with two widely-used reasoning patterns, i.e., multi-hop reasoning and multi-object reasoning. **Multi-hop reasoning** (e.g., the period of the author of *Off on a Comet* is 1828-1905) first infers an intermediate entity from the topic entity in the question (the author of *Off on a Comet* is Jules Verne), and then use it to infer the next intermediate entity until reaching the answer (the period of Jules Verne is 1828-1905). Therefore, the knowledge triples form a chain-like structure, where the tail entity of one triple is the head of the next one (e.g., Jules Verne). Given these triples, *Comp* discards all intermediate entities and concatenates other fields sequentially. **Multi-object reasoning** (e.g., the occupation of Jules Verne is novelist and playwright) infers several results from one topic entity, thus the knowledge triples share the same head entity and relation (Jules Verne and occupation). Given the triples, *Comp* discards the heads and relations expect the first one, and concatenates all tails with the first head and relation. Besides, our framework could also easily generalize to other reasoning patterns such as the comparative reasoning in the similar way by defining the sampling restrictions and *Comp* methods for triples. For each sentence we sample 2 to 3 triples matching the patterns.

4 EXPERIMENTS

4.1 Experimental Setup

4.1.1 Datasets. We use three KBQA datasets to evaluate KICP on knowledge-based reasoning, i.e., CN-QA (with CN-KG as KG), ComplexWebQuestions [35] and FreebaseQA [13] (both with Wikidata [37]), and a generative dataset Math23K [38] (with HowNet [32]) for generalization to other knowledge-related QA. The introduction and statistics of the datasets are available in Appendix A.

Table 1: Overall Results of All Methods on Four Datasets

Dataset	CN-QA		ComplexWebQuestions		FreebaseQA	Math23K
Metric	F1	EM	F1	EM	ACC	ACC
GPT4	0.459	0.358	0.802	0.721	**0.918**	/
ChatGLM2-6B	0.389	0.274	0.494	0.432	0.610	/
EmbedKGQA	0.417	0.303	0.760	0.730	0.707	/
BERT	0.607	0.458	0.856	0.763	0.896	0.801
RoBERTa	0.610	0.456	0.863	0.779	0.892	0.803
ERNIE	0.614	0.459	0.861	0.772	0.901	0.796
K-BERT	0.620	0.462	0.866	0.774	0.896	0.799
KEPLER	0.628	0.467	0.868	0.785	0.906	/
K-Adapter	0.612	0.462	0.866	0.802	0.905	/
KICP-KA	0.633	0.469	0.871	0.809	0.903	0.797
KICP-ATT	0.629	0.466	/	/	/	/
KICP	**0.639***	**0.480***	**0.880***	**0.819***	0.911*	**0.809***

KBQA answers questions with entities or attribute values from KG. To reduce computation complexity without losing much difficulty, we sample 10 hard candidate answers with the same type of the truth for prediction on KBQA. We also sample a sub-graph from the whole KG for each dataset to accelerate pretraining.

4.1.2 Baseline Methods. We compare KICP with original LMs **BERT** [6] and **RoBERTa** [22], and knowledge-enhanced LMs **ERNIE** [50], **K-BERT** [21], **KEPLER** [37] and **K-Adapter** [36]. We also include the embedding-based **EmbedKGQA** [33] and two LLMs **GPT4** [1] and **ChatGLM2** [8] as baselines for KBQA datasets. We provide a brief introduction to baselines in Appendix B.

4.1.3 Training Details. We implement KICP with Pytorch based on pretrained BERT by huggingface. [1] We use the "bert-base-chinese" version as *LM* on Chinese datasets CN-QA and Math23K, and "bert-base-uncased" on English datasets ComplexWebQuestions and FreebaseQA for all methods. The number of BERT layers of *Ad* for KA is 12 (equal to *LM*), the dimension is 384 for hidden vector (half of *LM*) and 768 for output vector(equal to *LM*).

We pretrain the model for 3 epochs with AdamW [24]. The batch size is set to 32, and the learning rate is 0.0005, which warms up over the first 10% steps, and then linearly decays. The masking probability for MLM is set to 0.15 in lesson 1 and 3, and 0.3 in lesson 2 as the corpus contains more repeated information.

We run all experiments on a Linux server with two 2.20 GHz Intel Xeon E5-2650 CPUs and a Tesla K80 GPU. [2]

4.2 Experimental Results

4.2.1 Overall Results. In this section, we compare KICP with all baselines. We use the F1 score (F1) and exact match score (EM) as metrics for multi-label datasets CN-QA and ComplexWebQuestions, and accuracy (ACC) for single-label dataset FreebaseQA. Math23K is evaluated with answer accuracy (ACC), i.e., the predicted expression is viewed correct if the computed answer equals the truth.

The results on four datasets are reported in Table 1. [3] We statistically test the improvement of KICP over baselines (except GPT4) with paired t-test, and find the improvement to be significant with $p < 0.05$ (marked *). We can get the following observations. First, KICP outperforms all baselines, which clearly demonstrates its effectiveness on knowledge learning and exploitation for QA tasks. Second, KICP performs better than K-Adapter with similar model but different pretraining task, showing the significant influence of pretraining task. Third, LLMs do not perform better than the fine-tuned methods on KBQA. GPT4 achieves comparable performance on the widely studied ComplexWebQuestions and FreebaseQA, but falls far behind on CN-QA, and the smaller ChatGLM2 performs even worse. Fourth, knowledge-enhanced methods outperform original LMs in most cases, proving that knowledge is a key element in QA reasoning. Last, knowledge injection does not bring much improvement and even negative effect on Math23K. The reason may be that Math23K requires NLU much more than knowledge.

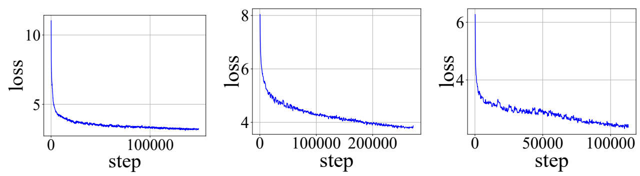
4.2.2 Ablation Study. In this section, we conduct ablation experiments to study the effectiveness of the attribute knowledge and knowledge adaptation (We will investigate the curriculum reasoning in detail in section 4.3). We introduce two variants of KICP: KICP-KA removes the knowledge adaptation module and directly trains the parameters of original LM, and KICP-ATT discards the attribute knowledge and pretrains only on the entity relation knowledge. The results of the two variants are also reported in Table 1. [4] We can summarize the following conclusions. First, the two variants perform worse than KICP, which shows that KA could reduce the negative impacts of generated corpus, and the attribute knowledge is also useful in KBQA. Next, in CN-QA, KICP-ATT performs worse than KICP-KA, which means that attribute knowledge exploitation contributes more than knowledge adaptation on this task. The result is reasonable since a large part of CN-QA requires attribute

[1] https://huggingface.co/transformers
[2] Our codes are available at https://github.com/l-xin/KICP.

[3] We do not evaluate KEPLER and K-Adapter on Math23K, as pretraining the two methods requires entity descriptions, which are unavailable on HowNet.
[4] The results of KICP-ATT on ComplexWebQuestions, FreebaseQA and Math23K are unavailable, as Wikidata and HowNet do not contain attribute knowledge.

Table 2: Performances on Easy and Hard Questions

Dataset	CN-QA		FreebaseQA	
Difficulty	Easy	Hard	Easy	Hard
BERT	0.633	0.603	0.920	0.891
KICP	0.676	0.634	0.933	0.907



(a) CN-KG (b) Wikidata (c) HowNet

Figure 3: Pretraining loss trend on three KGs in lesson 1.

knowledge (about 45%). Last, KICP-KA performs worse than BERT in Math23K, which may be due to that KICP-KA hurts the NLU ability of original LM in knowledge pretraining.

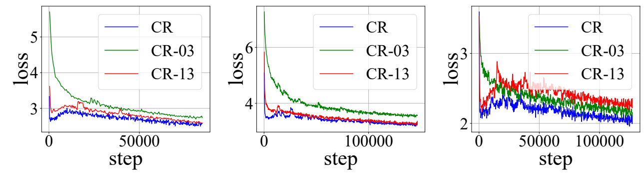
4.2.3 Performance over Difficulty. We also investigate the performance of KICP on questions with different difficulties to study the complex reasoning ability. We split CN-QA and FreebaseQA into easy questions (answerable with one knowledge triple) and hard ones (requiring multiple triples). [5] We report the performances of KICP and BERT in Table 2 (F1 on CN-QA and ACC on FreebaseQA). We have the following observations. First, it is reasonable that all methods perform much better on the easy questions than the hard ones. Second, KICP outperforms BERT on both easy and hard questions, showing that both easy and complex QA reasoning benefits from knowledge injection and exploitation. Next, the improvement on hard questions are larger in FreebaseQA. The reason may be that KICP are pretrained on corpus requiring more reasoning ability, which contributes to the higher performance in hard questions. However, in CN-QA the easy questions benefit more, which may result from the much larger proportion of easy questions benefiting from knowledge, and leads to a higher improvement.

4.3 Curriculum Reasoning Analysis

In this section, we investigate the feasibility and effectiveness of curriculum reasoning in KICP.

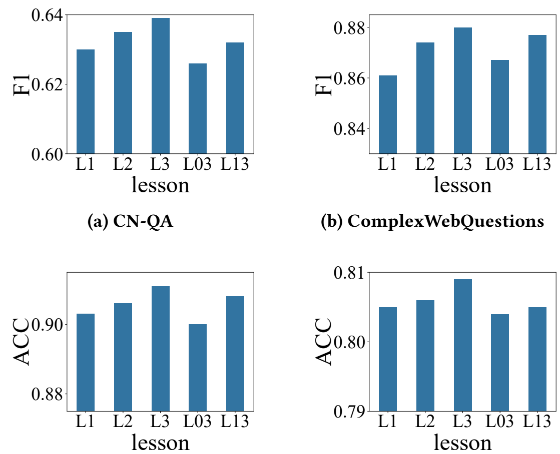
4.3.1 Loss of Curriculum Pretraining. Obviously the corpus generated by the CR module greatly differs from the natural ones. Therefore, to verify the feasibility of pretraining with such corpus, we plot the trend of loss in pretraining. Due to limited space, we report the lesson 1 results on three KGs in Figure 3. From the figure, the loss keeps dropping and then gradually converges, which demonstrates that the generated corpus contains enough information to train the LM for knowledge learning, although it may seem odd compared with natural ones.

[5] ComplexWebQuestions only contains hard questions and Math23K is a generative dataset which is hard to distinguish knowledge requirement, so we do not conduct the experiment on the two datasets.



(a) CN-KG (b) Wikidata (c) HowNet

Figure 4: Pretraining loss trend on three KGs in lesson 3.



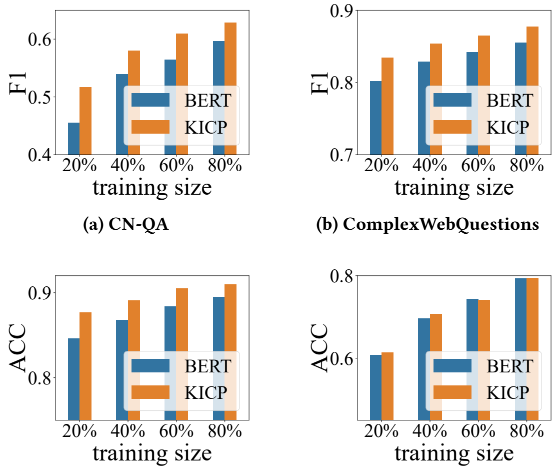
(a) CN-QA (b) ComplexWebQuestions

(c) FreebaseQA (d) Math23K

Figure 5: Performances of LM pretrained for each lesson.

CR also aims to reduce difficulty of pretraining LM for complex reasoning in lesson 3. To investigate the effectiveness, we plot the loss trend in lesson 3 in Figure 4 with two variants: CR-03 directly trains on lesson 3 without previous lessons, and CR-13 skips lesson 2. There are several observations. First, the loss of CR drops faster and finally reaches lower, proving that the curriculum setting could reduce the training difficulty. Second, the trend of CR-03 is similar to lesson 1 in Figure 3, meaning that in CR-03 the model may first learn basic knowledge as lesson 1 and then reasoning. Third, the loss of CR and CR-13 has a short increase in the beginning which may be due to the higher difficulty of lesson 3 and the different distribution from previous easier lesson. Last, CR-13 works better than CR-03 in CN-KG and Wikidata, showing that the LM can perform reasoning better with knowledge memorized. The exception in HowNet may be due to that HowNet mainly contains semantic information, which has been partially covered in LM.

4.3.2 Performance of Curriculum Reasoning. We also evaluate the effectiveness of CR on downstream QA tasks. Ideally, the LM performs better after pretrained on each lesson. Therefore, we evaluate the LM finishing lesson 1, 2, 3 ("L1", "L2", "L3") with CR-03 and CR-13 ("L03" and "L13") in Figure 5. We can get the following observations. First, performances of models keep increasing after finishing each lesson, which proves the above assumption. Second, L3 performs much better than L03 and L13 (all pretrained on lesson



(a) CN-QA **(b) ComplexWebQuestions**

(c) FreebaseQA **(d) Math23K**

Figure 6: Performances of KICP and BERT over training size.

3), showing that the curriculum setting helps in both convergence and the final outcome. Third, the results can also be viewed as an ablation study on each lesson ("L3" for "KICP", "L1" for "KICP w/o CR", "L13" for "KICP w/o L2", "L2" for "KICP w/o L3", and "L03" for "KICP w/o L1&L2"), which demonstrates the effectiveness of each lesson. Last, the performances on Math23K do not differ greatly. The reason may be that Math23K requires NLU more than knowledge, thus the effect of pretraining are limited.

4.4 Training Size Analysis

The pretrained LM aims to reduce the requirement of labeled data and improve the generalization, so the LM pretrained on the KG is expected to have a better performance than the original ones with limited labeled data. Therefore, we split the QA datasets with different training proportion (i.e., 20%, 40%, 60%, 80%) to evaluate performances of KICP and BERT. The results are demonstrated in Figure 6. From the figure, there are several observations. First, the performances of both KICP and BERT reasonably increase with more training samples. Next, although KICP outperforms BERT in all training settings, generally the differences are larger with less training data. The reason may be that the pretrained KICP could utilize the knowledge learned from KG and exploit less labeled data to learn the mapping from question to answer and achieve a good performance, while BERT needs to learn knowledge from the labeled data, which may be harder without enough data and result in worse performance.

4.5 Case Study

We demonstrate three typical cases by KICP and BERT on KBQA datasets in Table 3, and provide more in Appendix C. In case 1, BERT does not understand the knowledge about the lyricist of the song, and fails in the question, while KICP learns related knowledge in pretraining and correctly answer the question. In case 2, KICP is capable of conducting multi-hop reasoning to find the complex relation between "Thomas Harris", "*The Silence of the Lambs*" and

Table 3: Cases of KICP and BERT

Case 1: Who composed the song *Alexander's Ragtime Band* in 1911 ? **KICP:** Irving Berlin (**correct**) **BERT:** Woody Guthrie (**wrong**)
Case 2: Thomas Harris's 1988 novel *The Silence of the Lambs* was actually a sequel - what was the name of the first book in the series ? **KICP:** *Red Dragon* (**correct**) **BERT:** *Dubliners* (**wrong**)
Case 3: Which producer is responsible for *Pearl Harbour*, *Pirates of the Caribbean*, and *Armageddon* ? **KICP:** Robert Mulligan (**wrong**) **BERT:** John Ridley (**wrong**) **Answer:** Jerry Bruckheimer

"*Red Dragon*" for the answer when the direct relation is unavailable, while BERT does not support such complex reasoning. In case 3, although both methods fail in the question, KICP predicts a closer answer which is also a producer with related knowledge, but BERT fails and makes an unrelated prediction.

5 CONCLUSION

In this paper, we proposed a general Knowledge-Injected Curriculum Pretraining framework (KICP) to learn the KG for question answering, which could work with different detailed techniques for flexible application. We developed a general knowledge injection module to convert the KG into the pretraining corpus for LM with three key steps, and proposed a knowledge adaptation module to reduce the negative impacts of the gap between the generated and natural corpus by keeping the NLU ability of LM in knowledge learning. Furthermore, we designed a curriculum reasoning module to effectively pretrain the LM for human-like complex knowledge reasoning. Experimental results on four QA datasets demonstrated that the proposed KICP could achieve a more comprehensive learning and exploitation of KG for questions answering, and the curriculum setting could effectively reduce the pretraining difficulty and promote the outcome.

The proposed framework still had some limitations. First, the diversity of corpus generated by KICP was limited, and it would benefit if the generated corpus could be more similar to natural ones. Second, in the paper we mainly focused on the LM for language understanding, and we will generalize our framework to generative LM in the future. Last, KICP only exploited the KG as knowledge source, and there were much more types of knowledge to be studied.

ACKNOWLEDGMENTS

This research was partial supported by grants from the National Key Research and Development Program of China (2021YFF0901005) and the National Natural Science Foundation of China (62106244, U20A20229), and the University Synergy Innovation Program of Anhui Province (GXXT-2022-042), and OPPO Research Fund.

REFERENCES

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774* (2023).

[2] Oshin Agarwal, Heming Ge, Siamak Shakeri, and Rami Al-Rfou. 2021. Knowledge Graph Based Synthetic Corpus Generation for Knowledge-Enhanced Language Model Pre-training. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 3554–3565.

[3] Wenhu Chen, Yu Su, Xifeng Yan, and William Yang Wang. 2020. KGPT: Knowledge-Grounded Pre-Training for Data-to-Text Generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 8635–8648.

[4] Xiang Chen, Ningyu Zhang, Xin Xie, Shumin Deng, Yunzhi Yao, Chuanqi Tan, Fei Huang, Luo Si, and Huajun Chen. 2022. Knowprompt: Knowledge-aware prompt-tuning with synergistic optimization for relation extraction. In *Proceedings of the ACM Web conference 2022*. 2778–2788.

[5] Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, and Ziqing Yang. 2021. Pre-training with whole word masking for chinese bert. *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 29 (2021), 3504–3514.

[6] 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, Volume 1 (Long and Short Papers)*. 4171–4186.

[7] Ning Ding, Yujia Qin, Guang Yang, Fuchao Wei, Zonghan Yang, Yusheng Su, Shengding Hu, Yulin Chen, Chi-Min Chan, Weize Chen, et al. 2022. Delta tuning: A comprehensive study of parameter efficient methods for pre-trained language models. *arXiv preprint arXiv:2203.06904* (2022).

[8] Zhengxiao Du, Yujie Qian, Xiao Liu, Ming Ding, Jiezhong Qiu, Zhilin Yang, and Jie Tang. 2022. GLM: General Language Model Pretraining with Autoregressive Blank Infilling. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 320–335.

[9] Shangbin Feng, Vidhisha Balachandran, Yuyang Bai, and Yulia Tsvetkov. 2023. Factkb: Generalizable factuality evaluation using language models enhanced with factual knowledge. *arXiv preprint arXiv:2305.08281* (2023).

[10] Ziniu Hu, Yichong Xu, Wenhao Yu, Shuohang Wang, Ziyi Yang, Chenguang Zhu, Kai-Wei Chang, and Yizhou Sun. 2022. Empowering language models with knowledge graph reasoning for open-domain question answering. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. 9562–9581.

[11] Xiao Huang, Jingyuan Zhang, Dingcheng Li, and Ping Li. 2019. Knowledge graph embedding based question answering. In *Proceedings of the twelfth ACM international conference on web search and data mining*. 105–113.

[12] Zhenya Huang, Xin Lin, Hao Wang, Qi Liu, Enhong Chen, Jianhui Ma, Yu Su, and Wei Tong. 2021. Disenqnet: Disentangled representation learning for educational questions. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 696–704.

[13] Kelvin Jiang, Dekun Wu, and Hui Jiang. 2019. FreebaseQA: A New Factoid QA Data Set Matching Trivia-Style Question-Answer Pairs with Freebase. In *Proceedings of NAACL-HLT*. 318–323.

[14] Bill Yuchen Lin, Xinyue Chen, Jamin Chen, and Xiang Ren. 2019. KagNet: Knowledge-Aware Graph Networks for Commonsense Reasoning. 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)*. 2829–2839.

[15] Xin Lin, Zhenya Huang, Hongke Zhao, Enhong Chen, Qi Liu, Defu Lian, Xin Li, and Hao Wang. 2023. Learning Relation-Enhanced Hierarchical Solver for Math Word Problems. *IEEE Transactions on Neural Networks and Learning Systems* (2023).

[16] Jiayu Liu, Zhenya Huang, Xin Lin, Qi Liu, Jianhui Ma, and Enhong Chen. 2022. A cognitive solver with autonomously knowledge learning for reasoning mathematical answers. In *2022 IEEE International Conference on Data Mining (ICDM)*. IEEE, 269–278.

[17] Jiayu Liu, Zhenya Huang, Zhiyuan Ma, Qi Liu, Enhong Chen, Tianhuang Su, and Haifeng Liu. 2023. Guiding Mathematical Reasoning via Mastering Commonsense Formula Knowledge. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 1477–1488.

[18] Jiayu Liu, Zhenya Huang, Chengxiang Zhai, and Qi Liu. 2023. Learning by Applying: A General Framework for Mathematical Reasoning via Enhancing Explicit Knowledge Learning. *arXiv preprint arXiv:2302.05717* (2023).

[19] Lihui Liu, Yuzhong Chen, Mahashweta Das, Hao Yang, and Hanghang Tong. 2023. Knowledge Graph Question Answering with Ambiguous Query. In *Proceedings of the ACM Web Conference 2023*. 2477–2486.

[20] Linlin Liu, Xin Li, Ruidan He, Lidong Bing, Shafiq Joty, and Luo Si. 2022. Enhancing multilingual language model with massive multilingual knowledge triples. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. 6878–6890.

[21] Weijie Liu, Peng Zhou, Zhe Zhao, Zhiruo Wang, Qi Ju, Haotang Deng, and Ping Wang. 2020. K-bert: Enabling language representation with knowledge graph. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 2901–2908.

[22] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692* (2019).

[23] Robert Logan, Nelson F Liu, Matthew E Peters, Matt Gardner, and Sameer Singh. 2019. Barack's Wife Hillary: Using Knowledge Graphs for Fact-Aware Language Modeling. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. 5962–5971.

[24] Ilya Loshchilov and Frank Hutter. 2019. Decoupled Weight Decay Regularization. In *International Conference on Learning Representations*.

[25] Pan Lu, Liang Qiu, Kai-Wei Chang, Ying Nian Wu, Song-Chun Zhu, Tanmay Rajpurohit, Peter Clark, and Ashwin Kalyan. 2022. Dynamic prompt learning via policy gradient for semi-structured mathematical reasoning. *arXiv preprint arXiv:2209.14610* (2022).

[26] Yinquan Lu, Haonan Lu, Guirong Fu, and Qun Liu. 2022. KELM: Knowledge Enhanced Pre-Trained Language Representations with Message Passing on Hierarchical Relational Graphs. In *ICLR 2022 Workshop on Deep Learning on Graphs for Natural Language Processing*.

[27] Denis Lukovnikov, Asja Fischer, and Jens Lehmann. 2019. Pretrained Transformers for Simple Question Answering over Knowledge Graphs. In *The Semantic Web–ISWC 2019: 18th International Semantic Web Conference, Auckland, New Zealand, October 26–30, 2019, Proceedings, Part I*. 470–486.

[28] 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 *Proceedings of the AAAI conference on artificial intelligence*, Vol. 34. 8449–8456.

[29] Yu Meng, Yunyi Zhang, Jiaxin Huang, Yu Zhang, and Jiawei Han. 2022. Topic discovery via latent space clustering of pretrained language model representations. In *Proceedings of the ACM Web Conference 2022*. 3143–3152.

[30] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems* 35 (2022), 27730–27744.

[31] Matthew E Peters, Mark Neumann, Robert Logan, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A Smith. 2019. Knowledge Enhanced Contextual Word Representations. 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)*. 43–54.

[32] Fanchao Qi, Chenghao Yang, Zhiyuan Liu, Qiang Dong, Maosong Sun, and Zhendong Dong. 2019. OpenHowNet: An Open Sememe-based Lexical Knowledge Base. *arXiv preprint arXiv:1901.09957* (2019).

[33] Apoorv Saxena, Aditay Tripathi, and Partha Talukdar. 2020. Improving multi-hop question answering over knowledge graphs using knowledge base embeddings. In *Proceedings of the 58th annual meeting of the association for computational linguistics*. 4498–4507.

[34] Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xin Tian, Danxiang Zhu, Hao Tian, and Hua Wu. 2019. Ernie: Enhanced representation through knowledge integration. *arXiv preprint arXiv:1904.09223* (2019).

[35] Alon Talmor and Jonathan Berant. 2018. The Web as a Knowledge-Base for Answering Complex Questions. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. 641–651.

[36] Ruize Wang, Duyu Tang, Nan Duan, Zhongyu Wei, Xuan-Jing Huang, Jianshu Ji, Guihong Cao, Daxin Jiang, and Ming Zhou. 2021. K-Adapter: Infusing Knowledge into Pre-Trained Models with Adapters. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*. 1405–1418.

[37] Xiaozhi Wang, Tianyu Gao, Zhaocheng Zhu, Zhengyan Zhang, Zhiyuan Liu, Juanzi Li, and Jian Tang. 2021. KEPLER: A unified model for knowledge embedding and pre-trained language representation. *Transactions of the Association for Computational Linguistics* 9 (2021), 176–194.

[38] Yan Wang, Xiaojiang Liu, and Shuming Shi. 2017. Deep neural solver for math word problems. In *Proceedings of the 2017 conference on empirical methods in natural language processing*. 845–854.

[39] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems* 35 (2022), 24824–24837.

[40] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed H Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. In *Advances in NeurIPS*.

[41] Qinzhuo Wu, Qi Zhang, Jinlan Fu, and Xuan-Jing Huang. 2020. A knowledge-aware sequence-to-tree network for math word problem solving. In *Proceedings of the 2020 conference on empirical methods in natural language processing (EMNLP)*.

7137–7146.
[42] Wenhan Xiong, Jingfei Du, William Yang Wang, and Veselin Stoyanov. 2020. Pretrained Encyclopedia: Weakly Supervised Knowledge-Pretrained Language Model. In *International Conference on Learning Representations*.
[43] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. 2022. ReAct: Synergizing Reasoning and Acting in Language Models. In *The Eleventh International Conference on Learning Representations*.
[44] Michihiro Yasunaga, Antoine Bosselut, Hongyu Ren, Xikun Zhang, Christopher D Manning, Percy S Liang, and Jure Leskovec. 2022. Deep bidirectional language-knowledge graph pretraining. *Advances in Neural Information Processing Systems* 35 (2022), 37309–37323.
[45] Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. 2021. QA-GNN: Reasoning with Language Models and Knowledge Graphs for Question Answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 535–546.
[46] Hongbin Ye, Ningyu Zhang, Shumin Deng, Xiang Chen, Hui Chen, Feiyu Xiong, Xi Chen, and Huajun Chen. 2022. Ontology-enhanced Prompt-tuning for Few-shot Learning. In *Proceedings of the ACM Web Conference 2022*. 778–787.
[47] Donghan Yu, Chenguang Zhu, Yiming Yang, and Michael Zeng. 2022. Jaket: Joint pre-training of knowledge graph and language understanding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 36. 11630–11638.
[48] Wen Zhang, Yushan Zhu, Mingyang Chen, Yuxia Geng, Yufeng Huang, Yajing Xu, Wenting Song, and Huajun Chen. 2023. Structure Pretraining and Prompt Tuning for Knowledge Graph Transfer. In *Proceedings of the ACM Web Conference 2023*. 2581–2590.
[49] Xikun Zhang, Antoine Bosselut, Michihiro Yasunaga, Hongyu Ren, Percy Liang, Christopher D Manning, and Jure Leskovec. 2022. GreaseLM: Graph REASoning Enhanced Language Models. In *International conference on learning representations*.
[50] Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019. ERNIE: Enhanced Language Representation with Informative Entities. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. 1441–1451.
[51] Wayne Xin Zhao, Kun Zhou, Zheng Gong, Beichen Zhang, Yuanhang Zhou, Jing Sha, Zhigang Chen, Shijin Wang, Cong Liu, and Ji-Rong Wen. 2022. JiuZhang: A Chinese Pre-trained Language Model for Mathematical Problem Understanding. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 4571–4581.
[52] Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc V Le, et al. 2022. Least-to-Most Prompting Enables Complex Reasoning in Large Language Models. In *The Eleventh International Conference on Learning Representations*.
[53] Chenguang Zhu, Yichong Xu, Xiang Ren, Bill Yuchen Lin, Meng Jiang, and Wenhao Yu. 2023. Knowledge-augmented methods for natural language processing. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*. 1228–1231.

A DATASETS

CN-QA is a Chinese KBQA dataset collected from smart voice assistant accompanied by a KG named CN-KG with both entity relations and attributes. **ComplexWebQuestions** [35] is a public KBQA dataset with complex questions built on WebQuestions and Freebase. **FreebaseQA** [13] is another public KBQA dataset based on Freebase with both simple and complex questions derived from TriviaQA and trivia websites. Since Freebase has been merged to Wikidata, we use the Wikidata dump in [37], and map entities to Wikidata to construct an answerable subset for ComplexWebQuestions and FreebaseQA. **Math23K** [38] is a public generative math word problem dataset which answers the question with a generated mathematical expression. We construct a KG based on the semantic web HowNet [32] for Math23K following [41]. The statistics of the datasets are available in Table 4.

B INTRODUCTION TO BASELINES

The introduction to the baselines are listed as follows.

- **BERT** [6] was the most widely used pretrained language model, based on which our framework is implemented, thus we add BERT as baseline to evaluate the improvement.
- **RoBERTa** [22] studied the impacts of hyperparameters and task design in pretraining, and achieved a robustly optimized BERT with significant improvements.
- **ERNIE** [50] developed an aggregator network to explicitly combine the entity embedding learned from KG with the semantics learned by LM to inject knowledge into the LM.
- **K-BERT** [21] directly linked the related KG triples with the sentence to inject the knowledge, which was fed to the LM together for the knowledge-enhanced representation.
- **KEPLER** [37] trained the LM as the knowledge embedding model, where the entity embedding was generated by the LM on the entity description.
- **K-Adapter** [36] designed a neural adapter for each kind of infused knowledge, and trained the adapters on different knowledge pretraining tasks.
- **EmbedKGQA** [33] represented the question and KG in the same latent space, and inferred the answer with simple vector computation.
- **GPT4** [1] is the state-of-the-art LLMs developed by OpenAI, which provides API to access the service.
- **ChatGLM2** [8] is an open-sourced bilingual LLMs with good performance, with its 6B pretrained weights released.

C MORE CASES

We also provide more cases predicted by KICP and BERT on the KBQA datasets in Table 5 in addition to section 4.5. We classify these cases into three categories, i.e., the easy questions, hard questions, and wrong questions that both KICP and BERT fail. We can summarize the following observations. First, the easy questions can be answered with only one knowledge triples, which investigates whether the LM can memorize and exploit the knowledge. From the cases, KICP performs better than BERT. Next, the hard questions require reasoning over multiple knowledge facts. There are two typical mistakes in these cases, i.e., wrong answers (case 6 and 7) and failed prediction (case 5), which shows that the method may be not so capable of effective reasoning. Last, there are also questions mistakenly answered by KICP (case 8 and 9). In these cases, both the two methods make similar wrong prediction, which shows that there are still much room to improve for KICP, such as more reasoning patterns and more efficient knowledge learning and exploitation.

D SAMPLES OF CORPUS

We demonstrate some samples of the constructed corpora for the three lessons of the CR module in Table 6. We place the unmasked version of each sentence on first line and masked one on second, and recover the split words for readability. The sentences are all in lower cases due to tokenization. We also provide related knowledge in the last two lines for lesson 3 for readability as some key information may be discarded.

Table 4: Statistics of Datasets

Dataset KG	CN-QA CN-KG	ComplexWebQuestions Wikidata	FreebaseQA Wikidata	Math23K HowNet
#Questions	13,041	13,544	15,811	23,162
#Simple questions	12,265	0	13,070	/
#Hard questions	776	13,544	2,741	/
Avg. answer per question	1.67	1.43	1	1
#Entity	1,477,923	397,133	397,133	237,861
#Relations & attributes	1,112	733	733	6
#All triples	6,352,980	2,900,156	2,900,156	1,206,695
#Relation triples	4,081,756	2,900,156	2,900,156	1,206,695
#Attribute triples	2,271,224	0	0	0
# Corpus for lesson 1	6,352,980	2,900,156	2,900,156	1,206,695
# Corpus for lesson 2	1,806,861	3,128,153	3,128,153	1,356,960
# Corpus for lesson 3	1,806,861	3,128,153	3,128,153	1,356,960

Table 5: More Cases Predicted by KICP and BERT

Category	Cases
Easy	**Case 1:** Aberystwyth lies on which bay ? **KICP:** Cardigan (**correct**) **BERT:** Blaenau Gwent (**wrong**)
	Case 2: In *Alice in Wonderland*, who wanted to decapitate anyone who offended her ? **KICP:** Queen of Hearts (**correct**) **BERT:** Daisy Fay (**wrong**)
	Case 3: Who wrote the thriller novel *Birds of Prey* ? **KICP:** Wilbur Smith (**correct**) **BERT:** Ludwig von Mises (**wrong**)
	Case 4: Io, Europa, Ganymede and Callisto are all moons of which planet in our solar system ? **KICP:** Jupiter (**correct**) **BERT:** Pluto (**wrong**)
Hard	**Case 5:** What kind of money does the country with the nation anthem *Du gamla, Du fria* use ? **KICP:** Swedish Krona (**correct**) **BERT:** / (**wrong**)
	Case 6: What form of government is used in the country that uses Chilean Peso ? **KICP:** Presidential system | Unitary state (**correct**) **BERT:** Presidential system | Unitary state | Patrimonial monarchy (**wrong**)
	Case 7: What is the nationality of the author of *The Little Prince* ? **KICP:** France (**correct**) **BERT:** America (**wrong**)
Wrong	**Case 8:** Which comedy actor played Charlie Bind in the 1964 film *Carry on Spying* ? **KICP:** Peter Hinwood (**wrong**) **BERT:** Peter Hinwood (**wrong**) **Answer:** Charles Hawtrey
	Case 9: What team did Drogba play for that won the 2014 Coupe de France Final championship ? **KICP:** Piast Gliwice (**wrong**) **BERT:** Germinal Beerschot (**wrong**) **Answer:** En Avant de Guingamp

Table 6: Samples of the Constructed Corpus in the CR Module

Lesson		Samples
Lesson 1	(1)	[CLS] sir frederick ashton nationality united kindom [SEP] [CLS] [MASK] [MASK] [MASK] nationality united kindom [SEP]
	(2)	[CLS] wilhelm friedrich kuhne member of royal society [SEP] [CLS] wilhelm friedrich kuhne member of [MASK] [MASK] [SEP]
	(3)	[CLS] republic of maldives used money maldivian rufiyah [SEP] [CLS] republic of maldives [MASK] [MASK] maldivian rufiyah [SEP]
	(4)	[CLS] sarbogard district time euro time [SEP] [CLS] sarbogard district time [MASK] orthogonal [SEP]
	(5)	[CLS] first hellenic republic flag flag of greece [SEP] [CLS] [MASK] [MASK] [MASK] flag flag of greece [SEP]
Lesson 2	(6)	[CLS] collaroy plateau based in p : nsw [SEP] au - ns divides into gundagai shire council [SEP] collaroy plateau based in divides into gundagai shire council [SEP] [CLS] collaroy plateau based in p : nsw [SEP] au - ns [MASK] into gundagai shire council [SEP] collaroy plateau based in [MASK] [MASK] gundagai shire council [SEP]
	(7)	[CLS] star fox 64 3d part of the series star fox (virtual boy) [SEP] starfox (virtual boy) characters fox makuraudo [SEP] fox mccloud recording by ohara takashi [SEP] star fox 64 3d part of the series characters recording by ohara takashi [SEP] [CLS] star fox 64 3d part of the series [MASK] fox [MASK] [MASK] [MASK]) [SEP] starfox (virtual boy) [MASK] fox makuraudo [SEP] [MASK] [MASK] [MASK] [MASK] recording by ohara takashi [SEP] star fox 64 3d part of the series [MASK] recording by ohara takashi [SEP]
	(8)	[CLS] spannarhyttan timezone utc + 2 : 00 [SEP] spannarhyttan timezone utc + 1 : 00 [SEP] spannarhyttan timezone utc + 2 : 00 utc + 1 : 00 [SEP] [CLS] spannarhyttan timezone utc [MASK] [MASK] : [MASK] [SEP] spannarhyttan [MASK] [MASK] utc + 1 : 00 [SEP] spannarhyttan ##unes ##zone [MASK] [MASK] 133 : [MASK] utc + 1 : 00 [SEP]
Lesson 3	(9)	[CLS] theobald ziegler working at on lake the rhine [SEP] [CLS] theobald ziegler working at on lake [MASK] [MASK] [SEP] ([CLS] theobald ziegler working at strassbourg [SEP] [CLS] strassbourg on lake the rhine [SEP])
	(10)	[CLS] ferrieres , somme shares border with ailly - sur - somme pont - de - metz [SEP] [CLS] ferrieres , somme [MASK] [MASK] [MASK] ailly - sur - somme pont - de - metz [SEP] ([CLS] ferrieres , somme shares border with ailly - sur - somme [SEP] [CLS] ferrieres , somme shares border with pont - de - metz [SEP])