



# Improving and evaluating complex question answering over knowledge bases by constructing strongly supervised data

Xing Cao<sup>1,3</sup> · Yingsi Zhao<sup>2</sup> · Bo Shen<sup>1,3</sup>

Received: 2 March 2022 / Accepted: 17 October 2022 / Published online: 5 November 2022  
© The Author(s), under exclusive licence to Springer-Verlag London Ltd., part of Springer Nature 2022

## Abstract

Complex question answering (CQA) is widely used in real-world tasks such as search engines and intelligent customer service. With the development of large-scale knowledge bases, CQA over knowledge bases has attracted considerable attention in recent years. However, there are many types of complex questions, and few works deeply focus on the performance analysis of models for different types of questions. Another major challenge is the lack of complete supervised labels due to the expense of manual labelling, decreasing model interpretability and increasing the difficulty of model training. In this paper, we constructed a dataset, named *CoSuQue*, which includes multiple types of complex questions and complete supervised labels that are easily obtained. Our work provides an in-depth analysis of the model's ability to answer different types of questions, contributing a comprehensive evaluation of the performance of CQA models. Based on the ability of the model to handle different types of questions, the model structure can be improved in a more targeted manner. The different types of complex questions and the complete supervised labels allow the inference process of the model to be investigated. Furthermore, we propose a novel training method that leverages the proposed dataset to improve the performance of the model on other publicly available datasets. Experiments on the Complex WebQuestions and WebQuestionsSP datasets demonstrate the effectiveness of our approach on the CQA task.

**Keywords** Complex question answering · Knowledge bases · Complete supervised labels · Multiple question types

---

Yingsi Zhao and Bo Shen have contributed equally to this work.

---

✉ Yingsi Zhao  
yszhao@bjtu.edu.cn

Xing Cao  
caoxing@bjtu.edu.cn

Bo Shen  
bshen@bjtu.edu.cn

<sup>1</sup> School of Electronic and Information Engineering, Beijing Jiaotong University, Beijing 100044, China

<sup>2</sup> School of Economics and Management, Beijing Jiaotong University, Beijing 100044, China

<sup>3</sup> Key Laboratory of Communication and Information Systems, Beijing Municipal Commission of Education, Beijing 100044, China

## 1 Introduction

Natural language question answering is a critical artificial intelligence task that has attracted substantial attention in recent years. Question answering systems use two kinds of sources to determine an answer: unstructured text corpora [1–5] and knowledge bases [6–10]. Knowledge base question answering (KBQA) aims to determine the answer to a natural language question based on the facts available in a knowledge base (KB), such as Dbpedia [11], YAGO [12], and Freebase [13].

Because knowledge bases are large (for example, although Freebase is no longer updated, its scale is still on the order of hundreds of GB, and real-time access during model training is not feasible), traditional KBQA approaches usually adopt the following architecture: (1) **named entity recognition**, in which topic entities in complex questions are identified; (2) **entity linking**, in which the topic entity in the question is linked to a knowledge base; (3) **question-specific subgraph retrieval**, in which a subgraph corresponding to the question is constructed; and

(4) **entity scoring**, in which the subgraph entities are scored and sorted. In recent years, with the development of knowledge graph embedding and representation learning, the performance of information retrieval-based (IR-based) models, which encode input questions via a question representation module and output reasoning instructions, has improved considerably. However, the performance of these IR-based models is hindered by two major challenges.

**Challenge 1. A lack of supervised labels during each reasoning step.** In contrast to simple questions [14], answering complex questions requires the aggregation of more information and the use of multiple-reasoning steps (such as comparison, aggregation, and sorting); as a result, complex question answering is also known as multi-hop question answering. It is well known that obtaining the subquestions of complex questions or the subanswers of the results during each reasoning step is costly when a large amount of training data is required. Min et al. [15] manually labelled 400 examples with complete supervised labels to training their model. In addition to manual data annotation, deep reinforcement learning [16–18] is often used to address this problem when intermediate supervised data are not available. These deep reinforcement learning methods alleviate the delayed and sparse reward problem caused by weak supervision by designing new reward functions. He et al. [19] utilized the correspondence between the state information acquired from forwards and backwards reasoning processes to alleviate the problem of weak supervision. However, for some complex questions, the intermediate distributions of forwards and backwards reasoning differ.

**Challenge 2. Uninterpretable reasoning.** Compared with semantic parsing-based (SP-based) methods [20–22], which translate natural language questions into logical form expressions that can be executed directly on knowledge graphs, the working mechanisms of IR-based methods are less interpretable. One of the core components of IR-based models is a black-box style instruction module [19, 23, 24], which uses neural networks to parse complex questions and generate a sequence of reasoning instructions. While neural networks are powerful, the black-box style of the component results in a less interpretable inference process. Obviously, it is difficult to incorporate user interactions for further improvement. In addition, there are no datasets that include multiple types of complex questions, and each question type is clearly identified to analyse the strengths and weaknesses of the model.

Figure 1 shows part of the Freebase knowledge base. Two entities (dots) are connected by a relationship (line) representing a set of facts to form a triple, such as ⟨United States, currency, dollar⟩. The triple can be framed as a simple question: ‘What is the currency of the United

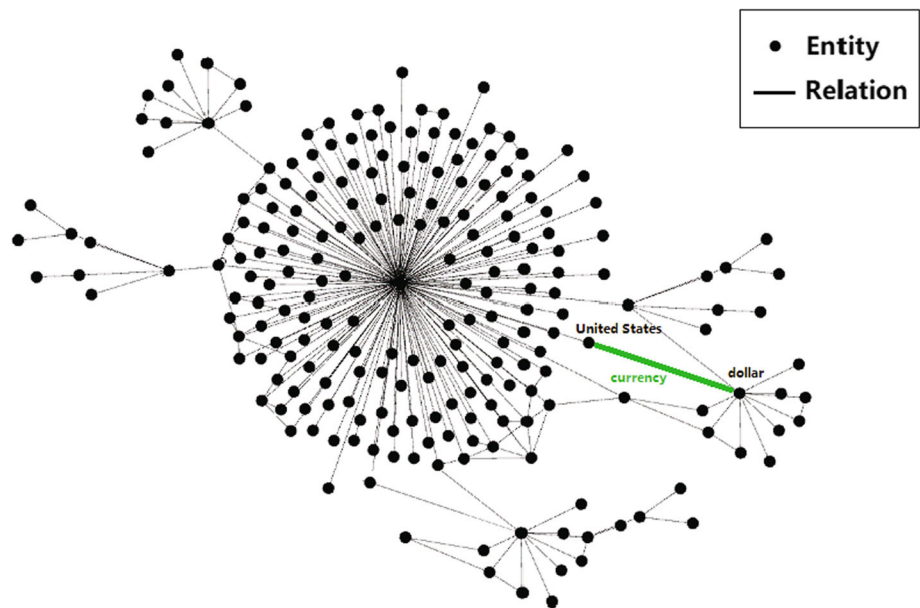
States?’. More complex questions can be constructed with different combinations of entities and relationships, and intermediate answers are readily available.

In this paper, to address the above challenges, we take advantage of the structured knowledge base to construct a dataset, *CoSuQue*, that contains multiple types of complex questions and includes all intermediate supervised labels in the inference process. We construct 9 types of questions with different combinations of logical operations to evaluate IR-based models. The impact of complete supervised labels on the training process and the performance of models in answering different questions are analysed in detail, and these experimental results contribute to explain the inference process of models. To take full advantage of the intermediate supervised labels in the *CoSuQue* dataset, we pretrained two models on the *CoSuQue* dataset and saved their parameters; then, we trained and tested these models on two public datasets, Complex WebQuestions (CWQ) [25] and WebQuestionsSP (WebQSP) [26]. The experimental results show that the convergence speed of the two models is significantly faster and the performance is also improved.

The contributions of this paper can be summarised as follows:

1. We propose a new dataset construction approach that includes steps such as query graph construction and natural language generation. The proposed approach is low cost and highly feasible. We construct a complex question answering dataset, *CoSuQue*, with strongly supervised labels that includes multiple types of complex questions and intermediate answers for each reasoning step.
2. An in-depth analysis of two typical IR-based models is performed on the proposed dataset. We analyse the impact of strongly supervised labels on model training in the context of different types of complex questions, as well as the preference of the two models for different types of complex questions. For most complex question answering systems, testing the ability of the system to answer different types of questions separately may help researchers to improve the model structure in a more targeted manner.
3. We propose a novel training method that uses our constructed dataset to improve model performance on other publicly available datasets. We pretrain IR-based models on the *CoSuQue* dataset, allowing the model to benefit from the complete supervised labels. Then, we test the model on two public complex question answering datasets, Complex WebQuestions and WebQuestionsSP. The results show that the convergence speed and accuracy of the model on the two

**Fig. 1** Part of the knowledge facts in Freebase. The black dots represent entities, and the lines represent relationships between entities



public datasets improve after the model is pretrained on the *CoSuQue* dataset.

Our code and interactive demo are publicly available at <https://github.com/NLPercx/CoSuQue>.

## 2 Related work

**Neural approaches for KBQA.** In this paper, we focus on complex question answering over knowledge bases. In addition to traditional methods, such as defining templates and rules, KBQA solutions can be divided into two main branches: SP-based methods and IR-based methods [27]. SP-based methods analyse natural language questions and generate logical query graphs corresponding to the questions (such as SPARQL and SQL). The generated query is executed over the given knowledge base to finally arrive at the answer. Lan et al. [28] consider two types of complex questions (with constraints and with multiple relation hops.) at the same time. The constraints were added during query generation to reduce the search space of the query graph. By designing different tasks (Split, TextSpanPrediction, HeadwordIdentification, and AttachmentRelationClassification) for query graph construction [29], the prior knowledge of BERT [30] can be used to improve the semantic parsing accuracy. However, the query graph generated by the model does not match the structure of the corresponding question, which increases the noise. To solve this problem, Chen et al. [21] employed abstract query graph (AQG) to describe the query structure. Das et al. [31] proposed a case-based reasoning model, which maintains a memory module that stores questions that have been answered correctly and a reasoning module that

generates logical forms by retrieving relevant cases from memory. The IR-based methods uses named entity tools to extract topic entities in the question and retrieve question-specific graphs from a set of knowledge graphs based on the topic entity. Finally, ranking algorithms are applied to select entities from the top position. To address issues with error cascades caused by the pipeline architecture, Zhang et al. [32] proposed an end-to-end variational learning algorithm that simultaneously handled uncertain topic entities and multi-hop reasoning. Yan et al. [33] identified all the paths in the subgraph that connected the topic entity to the candidate entity  $e_i$ , constructed the textual form of each path by replacing nodes with entity names and edges with relational names, and concatenated the question and path as an input sample for BERT. Finally, a score  $s_i$  was calculated for each candidate entity that indicated whether  $e_i$  is the answer entity. This approach solves the problem that the model only grasps the topological structure of the knowledge graph while ignoring the textual information.

The lack of supervised data for answers at each reasoning step remains a major challenge. He et al. [19] proposed a teacher-student structure that explores in both directions, allowing the two reasoning processes to synchronize with each other at intermediate steps. Qiu et al. [17] adopted a stepwise reasoning method based on reinforcement learning and proposed a potential-based reward shaping strategy to accelerate the convergence of the training algorithm.

**Knowledge graph embeddings.** Some researchers have built deep architectures to embed knowledge bases and represent entities and relations as low dimensional vectors in a continuous vector space, such as TransE [34], TransH

[35], TransR [36], TransD [37], LineaRE [38], and PairRE [39]. Saxena et al. [40] first used knowledge base embeddings for KBQA. They employed the pretrained model Roberta to encode complex questions as a relation vector  $e_q$ . This relation vector forms a triple with the head entity  $e_h$  and tail entity  $e_t$ , and the ComplEx score [41] was used to identify the answer entity. Reasoning plays a major role in question answering tasks. Ren et al. [42] proposed the Query2Box embedding-based framework, which regards first-order logical queries as geometric operations, and the resulting entity embedding is more suitable for question answering reasoning tasks. The NewLook method proposed by Liu et al. [43] used a nonlinear neural network to learn the projection operation. The cone embedding approach [44] represented entities and queries as Cartesian products of two-dimensional cones, instead of representing entities as points and questions as boxes, as performed in in Query2box.

### 3 Task definition

In this section, we first define knowledge bases and complex questions; then, we discuss the goal of the knowledge base question answering task. Table 1 shows the main notations used throughout this paper.

**Definition 1 (Knowledge Base/Graph)** A knowledge base  $G$  consists of an entity set  $E$ , a relation set  $R$ , and a set of knowledge facts  $K$  in the form of triples, denoted as  $G = \{ \langle h, r, t \rangle \in K \mid h, t \in E, r \in R \}$ . The *head entity*,  $h$ , denotes the source entity.  $r$  represents the relation, and  $t$  is the target entity, which is also called the *tail entity*. A triple  $\langle h, r, t \rangle$  denotes that a relationship  $r$  exists between the

head entity  $h$  and tail entity  $t$  and that this relationship is directional. Consider an example in which entity  $h$  is described as character *James Bond*, and entity  $t$  is the actor *Ian Fleming*. Then, a fact in the knowledge graph can be defined as  $\langle \text{James Bond}, \text{fictional\_universe.fictional\_universe.created\_by}, \text{Ian Fleming} \rangle$ , where the corresponding  $r$  is *fictional\\_universe.fictional\\_universe.created\\_by*.

**Definition 2 (Complex Question)** Questions that require multi-hop reasoning are called complex questions. Complex questions are more suitable for practical application scenarios than simple questions. For example, answering the question “Which movie was produced by Neil Moritz and starred Tupac Shakur?” requires that the model perform two-hop reasoning.

Given a question  $Q = \{q_1, q_2, q_3 \dots q_n\}$  consisting of  $n$  words, the task is defined as determining the answer to the given question using the facts  $K$  stored in the knowledge base  $G$ . Specifically, the goal of the knowledge base question answering task is to identify the answer entities in a knowledge base. In this paper, we focus on complex question answering, and there are often multiple triples between the answer entity and the topic entity in the knowledge base; thus, a question answering model should be able to learn multi-hop reasoning from question-answer pairs.

## 4 Proposed approach

In this section, we first introduce the definition and construction methods of the query graph in Sect. 4.1, which can be regarded as the skeleton of the complex question. Sect. 4.2 describes how to generate the corresponding natural language questions based on the query graph. In Sect. 4.3, we introduce two models: GraftNet, a classical IR-based model, and NSM, the model that achieved the best performance on the CWQ dataset. The NSM and GraftNet models are pretrained on our constructed dataset in Sect. 4.4.

### 4.1 Query graph construction

Constructing question answering datasets from unstructured textual data (such as Wikipedia) is expensive, especially when the complex question answering dataset requires multi-hop supervised data labelling. In contrast to unstructured data, knowledge bases contain structured data that is composed of a series of triples. Therefore, it is easy to obtain query graphs of different topological structures composed of multiple logical operations from the knowledge base.

**Table 1** The important symbols and their definitions

$Q$	The questions that need to be answered
$G$	The knowledge base
$G_q$	The question-specific subgraph
$E$	Entity set
$R$	Relation set
$K$	Knowledge facts
$h$	Head entity
$r$	Relation
$t$	Tail entity
$c$	Chain logical operation
$i$	Interaction logical operation
$e_n$	Entity
$e_r^n$	Answer entity set
$N_r(e)$	The neighbourhood entities of $e$

Complex questions include two common types of logical operations, *chain* and *interaction*, which are defined below.

- **Chain:** Consider an entity  $e \in E$ , and a set of relations  $r \in R$ . We can obtain a set  $e_r$ , where  $e_r = e' \in E \langle e, r, e' \rangle \in K$ .
- **Interaction:** Consider an entity  $e_1 \in E$ , an entity  $e_2 \in E$ , and a set of relations  $r \in R$ . We can obtain two sets:  $e_r^1 = \{e' \in E, \langle e_1, r, e' \rangle \in K\}$  and  $e_r^2 = \{e'' \in E, \langle e_2, r, e'' \rangle \in K\}$  by performing chain logical operation. The interaction operator obtains  $e_r^1 \cap e_r^2$ .

We perform different permutations and combinations of the above two logical operations to construct 9 types of query graphs  $\{1c, 2c, 3c, 2i, 2i^2, 1c2i, 2c2i, 2i1c \text{ and } 2i2c\}$ . These 9 query graphs correspond to 9 different types of natural language questions.

As shown in Fig. 2, the light blue circles represent the topic entities extracted from the question, and the blue and green circles represent the intermediate entities, which are also known as subanswers to the question. The yellow circle represents the answer entity. We refer to the topic entity as the head entity and perform a one-step *chain* logical operation to obtain  $e_{\text{answer}_{1c}}$ . The one-step *chain* logical operation  $1c$  can be defined with the following formula:

$$1c = \{e_{\text{answer}_{1c}} \in E, \langle \text{head}, r, e_{\text{answer}_{1c}} \rangle \in K\} \tag{1}$$

On the basis of  $e_{\text{answer}_{1c}}$ , another chain logical operation is performed, which is called a two-step chain operation. The answer to  $1c$  is the head entity of the second chain logical operation.  $2c$  represents the two-step chain operation, and  $e_{\text{answer}_{2c}}$  is the target entity. The two-step chain logical operation can be defined with the following formula:

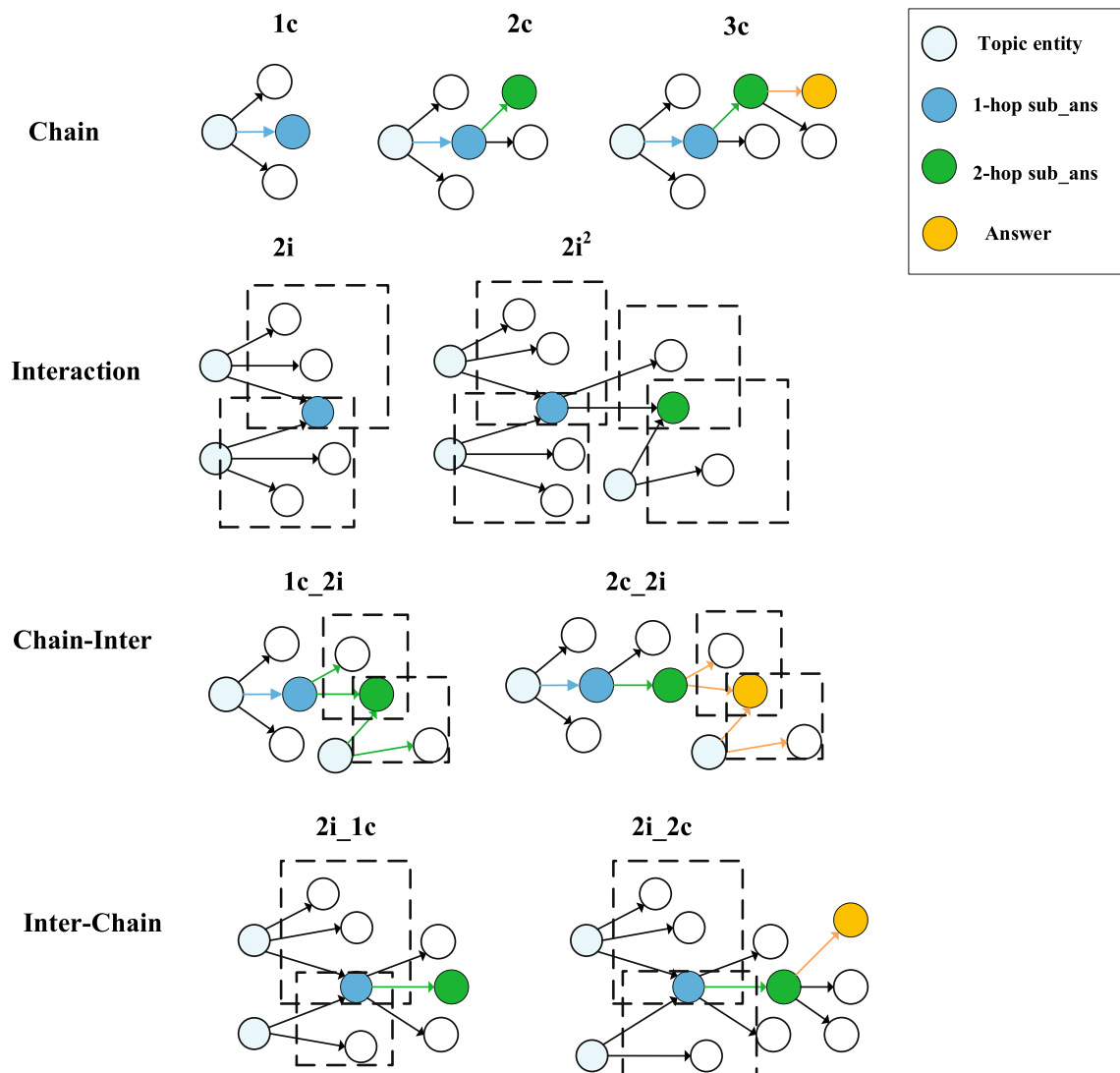


Fig. 2 Query graph



$$2c = \{e_{\text{answer}_{2c}} \in E, \langle \text{head}, r, e_{\text{answer}_{1c}} \rangle, \langle e_{\text{answer}_{1c}}, r, e_{\text{answer}_{2c}} \rangle \in K\} \tag{2}$$

Moreover,  $3c$  is obtained by adding a chain operation on the basis of  $2c$ , and the answer to  $2c$  is the head entity of the third chain operation.  $e_{\text{answer}_{3c}}$  is the target entity of  $3c$ . The three-step chain logical operation can be defined with the following formula:

$$3c = \{e_{\text{answer}_{3c}} \in E, \langle \text{head}, r, e_{\text{answer}_{1c}} \rangle, \langle e_{\text{answer}_{1c}}, r, e_{\text{answer}_{2c}} \rangle \in K, \langle e_{\text{answer}_{2c}}, r, e_{\text{answer}_{3c}} \rangle \in K\} \tag{3}$$

$2i$  represents the one-step interaction logical operation, which selects two head entities to perform chain logical operations, yielding two entity sets,  $e_r^1$  and  $e_r^2$ . The intersection of the two sets is used to obtain the final answer, namely  $e_{\text{answer}_{2i}} = e_r^1 \cap e_r^2$ . The one-step interaction logical operation  $2i$  can be defined with the following formula:

$$2i = \{e_{\text{answer}_{2i}} \in E, \langle \text{head}_1, r, e_r^1 \rangle, \langle \text{head}_2, r, e_r^2 \rangle \in K, e_r^1 \cap e_r^2\} \tag{4}$$

$2i^2$  represents the two-step interaction logical operation, which uses the answer to  $2i$  as a head entity and performs a one-step interaction operation with the other head entity  $\text{head}_3$  to determine the target entities  $e_{\text{answer}_{2i^2}}$ . The two-step interaction logical operation  $2i^2$  can be defined with the following formula:

$$2i^2 = \{e_{\text{answer}_{2i^2}} \in E, \langle \text{head}_1, r, e_r^1 \rangle, \langle \text{head}_2, r, e_r^2 \rangle, \langle e_{\text{answer}_{2i}}, r, e_r^3 \rangle, \langle \text{head}_3, r, e_r^4 \rangle \in K, e_r^3 \cap e_r^4\} \tag{5}$$

Increasing the semantic complexity of natural language questions is more suitable for practical application scenarios. We combine the two logical operations, chain and interaction, to construct more types of complex questions. The type of complex question in which there is a one-step interaction operation after the one-step chain operation is denoted as  $1c2i$ . After the chain operation is performed on the first head entity  $\text{head}_1$ , the obtained subanswer is used as one of the head entities in the next interaction operation.  $1c2i$  can be calculated as follows:

$$1c2i = \{e_{\text{answer}_{1c2i}} \in E, \langle \text{head}_1, r, e_{\text{answer}_{1c}} \rangle, \langle e_{\text{answer}_{1c}}, r, e_r^1 \rangle, \langle \text{head}_2, r, e_r^2 \rangle \in K, e_r^1 \cap e_r^2\} \tag{6}$$

Similarly, the type of complex question in which a two-step chain operation is followed by a one-step interaction operation is denoted as  $2c2i$ . The intermediate answers in this reasoning process include  $e_{\text{answer}_{1c}}$  and  $e_{\text{answer}_{2c}} \cdot e_{\text{answer}_{2c}}$

and  $\text{head}_2$  are the head entities of the interaction logical operation.  $2c2i$  can be calculated as follows:

$$2c2i = \{e_{\text{answer}_{2c2i}} \in E, \langle \text{head}_1, r, e_{\text{answer}_{1c}} \rangle, \langle e_{\text{answer}_{1c}}, r, e_{\text{answer}_{2c}} \rangle, \langle e_{\text{answer}_{2c}}, r, e_r^1 \rangle, \langle \text{head}_2, r, e_r^2 \rangle \in K, e_r^1 \cap e_r^2\} \tag{7}$$

Next, we reverse the order of the two types of logical operations; that is, the one-step interaction operation is first performed on two head entities to obtain  $e_{\text{answer}_{2i}}$ , and then, the one-step chain operation is performed on the obtained subanswer. This type of complex question is denoted as  $2i1c$ .  $2i1c$  can be calculated as follows:

$$2i1c = \{e_{\text{answer}_{2i1c}} \in E, \langle \text{head}_1, r, e_r^1 \rangle, \langle \text{head}_2, r, e_r^2 \rangle, \langle e_{\text{answer}_{2i}}, r, e_{\text{answer}_{2i1c}} \rangle \in K\} \tag{8}$$

Similarly, the intermediate answer entity of  $2i$  goes through two-step chain operations to obtain a query graph of type  $2i2c$ . Type  $2i2c$  includes two intermediate answers,  $e_{\text{answer}_{2i}}$  and  $e_{\text{answer}_{2i1c}}$ , and can be calculated as follows:

$$2i2c = \{e_{\text{answer}_{2i2c}} \in E, \langle \text{head}_1, r, e_r^1 \rangle, \langle \text{head}_2, r, e_r^2 \rangle, \langle e_{\text{answer}_{2i}}, r, e_{\text{answer}_{2i1c}} \rangle, \langle e_{\text{answer}_{2i1c}}, r, e_{\text{answer}_{2i2c}} \rangle \in K\} \tag{9}$$

This query graph construction method has several advantages. First, the process of constructing data is efficient and has a low cost. Second, the intermediate answer (fully supervised data) of the complex reasoning process is easy to obtain and has a high accuracy rate. More importantly, the intermediate supervised data in the reasoning process are crucial for training the model, especially for complex reasoning tasks.

### 4.2 Natural language question construction

The inputs to the complex question answering over knowledge bases model are natural language questions. After the various query graphs are constructed, the query graphs must be converted into the corresponding natural language questions.

For each question, we constructed a series of components, denoted as ‘A’, ‘B’, ‘R’, ‘C’ and the head entity. ‘A’ represents interrogative words, such as *what*, *which*, *who* and *where*. We use the Stanford CoreNLP<sup>1</sup> grammar tool to perform named entity recognition on the final answers to the complex questions. If the answer is a person’s name, ‘A’ chooses the word *Who*. If the answer is the name of an institution, a city, or a country, ‘A’ chooses among the words *where*, *what organization* and *which country*. ‘B’

<sup>1</sup> <https://stanfordnlp.github.io/CoreNLP/>.

represents linking verbs and depends on the number of final answer entities. If the number of answer entities is greater than one, ‘B’ selects a plural form, such as *are* or *were*. ‘R’ represents the relation set  $\{r_1, r_2, \dots, r_n\}$  connected to the head entity in the query graph. If  $r_n$  contains prepositions such as  $\{by, for, in, on, of, with\}$ , the preposition *of* is not added between the relation and the head entity. The head entity is as defined in Sect. 4.1 ‘C’ is a set of pronouns or phrases that connect the answers of the interaction operations, such as *that, the people who, the cities that, the organization that* and *in the month when*. The specific form of ‘C’ depends on the type of the answer of the interaction logical operation. For example, in the question corresponding to *2i1c* in Table 2, the answer of *2i* is a person’s name, then ‘C’ selects the phrase “*the people who*”. Specific templates are shown in Table 2.

### 4.3 Model and loss function

We chose two mainstream models, the GraftNet model [24] and the NSM model [19], to analyse and improve through the dataset constructed in this paper. The structure of the model and our improved loss function are described below.

#### 4.3.1 GraftNet

GraftNet is a conventional model for the KBQA task that combines a knowledge base with additional text to build hierarchical graphs and perform multi-hop reasoning. In this paper, we focus on the case in which the model uses

only the knowledge graph for question answering, with the goal of predicting whether an entity  $v$  in the knowledge base is the answer to question  $Q$ .

Figure 3 shows the architecture of the GraftNet model. Each word in the natural language question  $Q = \{q_1, q_2, q_3, \dots, q_n\}$  is converted to a Glove word embedding representation and encoded by a bidirectional long short-term memory (LSTM) network to obtain a set of hidden states  $\{h_j\}_{j=1}^n$ . The final state in the output of the LSTM is considered to be the initial representation of the question,  $h_q^{(0)}$ . The question initial representation is computed as:

$$h_q^{(0)} = LSTM(q_1, q_2, \dots, q_n)_{|n|} \tag{10}$$

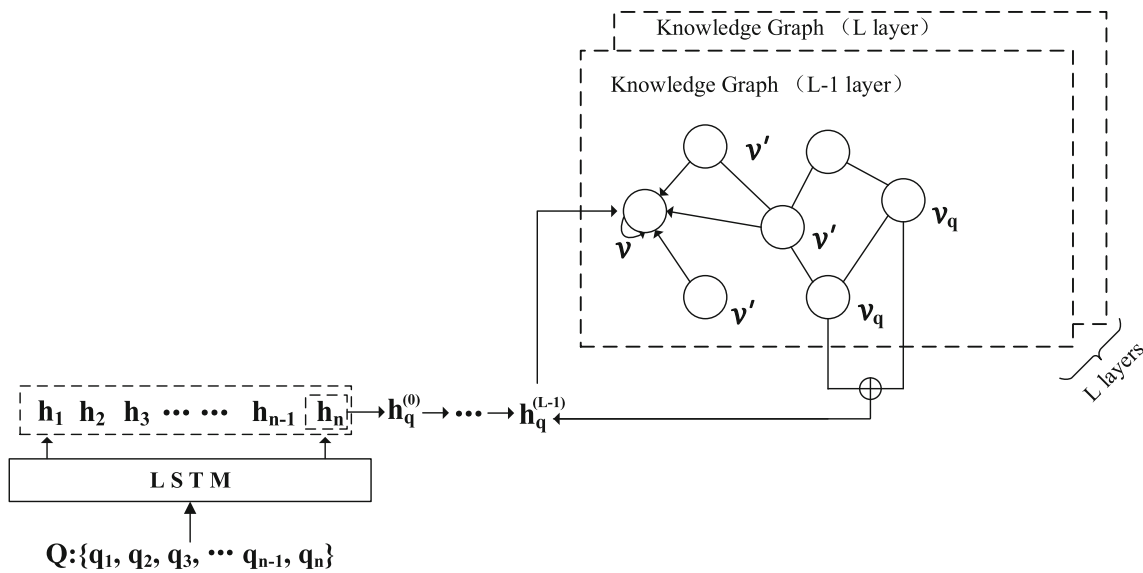
The GraftNet model updates node representations through message propagation and aggregation among entities in the knowledge graph, and the entity nodes are updated with a single-layer feedforward network (FFN) over the concatenation of three states:

$$h_v^{(l)} = FFN \left( \left[ h_v^{(l-1)}; h_q^{(l-1)}; \sum_r \sum_{v' \in N_r(v)} \alpha_r^{v'} \phi_r(h_{v'}^{(l-1)}) \right] \right) \tag{11}$$

where  $[\cdot]$  represents vector concatenation across rows, and  $h_v^{(l-1)}$  is the representation of entity  $v$  in the previous layer.  $h_q^{(l-1)}$  corresponds to the question representation in the  $(l - 1)$ -th layer, which can be calculated as follows:

**Table 2** Rules for generating natural language questions from query graphs

Question type	Example template	Example question
1 chain	A+B+R+{head}?	What is the currency of the United States?
2 chain	A+B+R1+R2+{head}?	What is the colour of currency of the United States?
3 chain	A+B+R1+R2+R3+{head}?	What is the French of the colour of currency of the United States?
2i interaction	(1)A+B+R1+{head1}+'and'+R2+{head2}? (2)A+B+R+{head1}+'and'+{head2}?	(1) Which country is participating countries of 2010 Winter Olympics and claimed of Gasherbrum IV? (2) What is the profession of Blu Mankuma and Luther Ingram?
2i <sup>2</sup> interaction	A+B+R1+{head1}+'and'+R2+C+(R3+{head2}+'and'+R4+{head3})?	What are the subjects of Etter Rubicon and events of the country that the geographic scope of Cuban Workers' Solidarity and contained by Cayo Megano?
1c2i	A+B+R1+{head1}+'and'+R2+R3+{head2}?	What are the affected areas of Hurricane Isabel and states provinces within of country of film Roller Boogie?
2c2i	A+B+R1+{head1}+'and'+R2+R3+R4+{head2}?	What is the country that nationality of Burnet R. Maybank and administrative parent of states provinces within of nationality of Hamilton Jordan?
2i1c	A+B+R1+C+R2+{head1}+'and'+R3+{head2}?	What are the professions of people who are associated authors of Feminist science fiction and associated authors of Science Fiction?
2i2c	A+B+R1+R2+C+R3+{head1}+'and'+R4+{head2}?	What were reviewed of the music of that film of Jordan Cronenweth and the film of Stuart Baird?



**Fig. 3** Architecture of the GraftNet model

$$h_q^{(l-1)} = FFN \left( \sum_{v_q \in S_q} h_{v_q}^{(l-1)} \right) \quad (12)$$

where  $S_q$  denotes the seed entities mentioned in the question. The third term in Eq. 11 aggregates the states from the entity neighbours of the current node.  $N_r(v)$  represents the entity neighbourhood, denoting the neighbours of  $v$  along edges of type  $r$ .  $\alpha_r^v$  is the attention weight of the question  $h_q^{(l-1)}$  and the set of relations connected to entity  $v$ , which can be computed as:

$$\alpha_r^v = \text{softmax} \left( x_r^T h_q^{(l-1)} \right) \quad (13)$$

$\varphi_r$  corresponds to relation specific transformations, and the update along a relation can be computed as:

$$\varphi_r \left( h_v^{(l-1)} \right) = pr_v^{(l-1)} FFN \left( x_r, h_v^{(l-1)} \right) \quad (14)$$

'PageRank' scores [45]  $pr_v^{(l)}$  which calculates the total weight of paths between a seed entity and the current node, as follows:

$$pr_v^{(0)} = \begin{cases} \frac{1}{|S_q|} & \text{if } v \in S_q \\ 0 & \text{o.w.} \end{cases} \quad (15)$$

$$pr_v^{(l)} = (1 - \lambda) pr_v^{(l-1)} + \lambda \sum_{v' \in N_r(v)} \alpha_r^{v'} pr_{v'}^{(l-1)} \quad (16)$$

The final representations  $h_v^{(L)}$  are used for binary classification to select the answers, as follows:

$$P_v(v \in \{a\}_q \| G_q, Q) = \sigma \left( W^T h_v^{(L)} + b \right) \quad (17)$$

where  $\sigma$  is the sigmoid function. The training process uses the binary cross-entropy loss over these probabilities.

$$\mathcal{L}_v = -(y \cdot \log P_v + (1 - y) \cdot \log(1 - P_v)) \quad (18)$$

where  $y$  is the ground truth label, which indicates whether  $v$  is the answer entity of the question. Our constructed dataset uses supervised labels in the intermediate inference steps, and the loss function of the GraftNet model trained with strongly supervised data is defined as follows:

$$\mathcal{L}_s^{\text{GraftNet}} = \sum_{l=1}^L -(y^l \cdot \log P^l + (1 - y^l) \cdot \log(1 - P^l)) \quad (19)$$

where  $L$  is the number of layers in the model, corresponding to the maximum path length that information should be propagated in the graph.

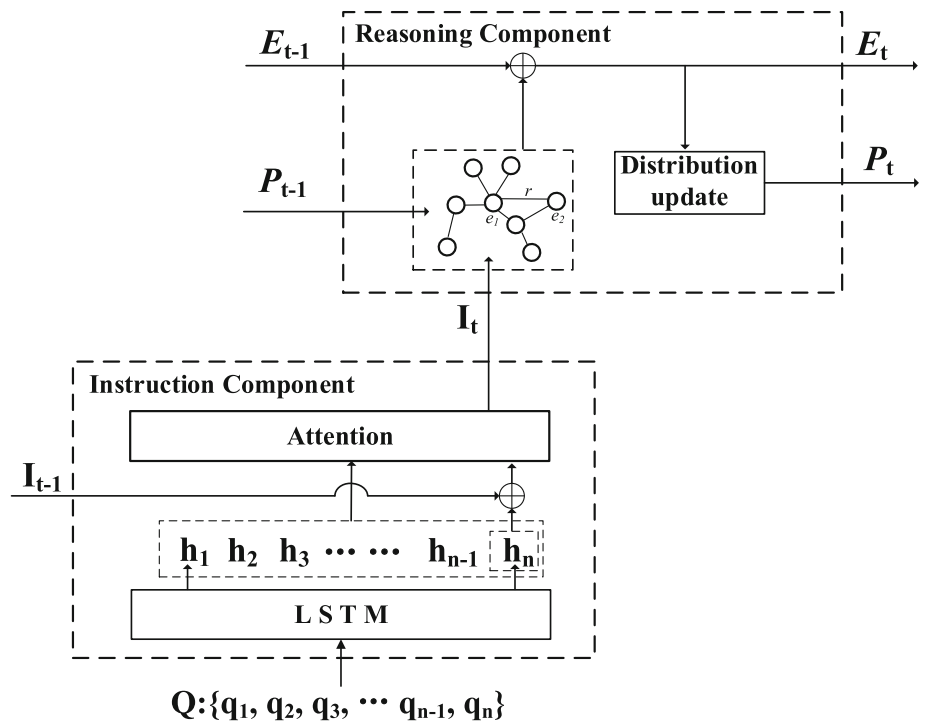
#### 4.3.2 NSM

The NSM model is the current best performing model on the CWQ dataset, which consists of two parts, an instruction component and a reasoning component. The output of the instruction module is used as the input to the reasoning module, which updates the entity representation in the subgraph and sorts the entities to determine the answer to the question.

The instruction component is used to analyse complex questions and generate a series of instruction vectors. As shown in Fig. 4, the input of the instruction component consists of a query embedding and an instruction vector



**Fig. 4** Architecture of the NSM model



obtained from the previous reasoning step. The question encoded by the bidirectional LSTM, a set of hidden states  $\{h_j\}_{j=1}^n$  is obtained, and the last hidden state is considered to be the question representation. The formula is as follows:

$$i^{(k)} = \sum_{j=1}^l \alpha_j^{(k)} h_j \tag{20}$$

$$\alpha_j^{(k)} = \text{softmax}_j \left( W_\alpha (q^{(k)} \odot h_j) + b_\alpha \right) \tag{21}$$

$$q^{(k)} = W^{(k)} [i^{(k-1)}; q] + b^{(k)} \tag{22}$$

where  $W_\alpha \in \mathbb{R}^{d \times 2d}$ ,  $W^{(k)} \in \mathbb{R}^{d \times d}$ ,  $b_\alpha \in \mathbb{R}^d$  and  $b^{(k)}$  are parameters to learn.

The instruction component is used to generate a series of instructions, and the reasoning component updates the representations of entity nodes in the subgraph according to these instructions. The initial entity embeddings are obtained as follows:

$$e^{(0)} = \sigma \left( \sum_{\langle e,r,e' \rangle \in N_r(e)} r \cdot W_T \right) \tag{23}$$

where  $W_T \in \mathbb{R}^{d_r \times d_r}$  are the parameters to learn, and  $N_r(e)$  means entity neighbourhood, denoting the neighbours of  $e$ . The output of the reasoning component is calculated as follows:

$$m_{\langle e,r,e' \rangle}^{(k)} = \sigma \left( i^{(k)} \odot W_R r \right) \tag{24}$$

$$\tilde{e}^{(k)} = \sum_{\langle e,r,e' \rangle \in N_e} P_{e'}^{(k-1)} \cdot m_{\langle e,r,e' \rangle}^{(k)} \tag{25}$$

$$e^{(k)} = FFN \left( [e^{(k-1)}; \tilde{e}^{(k)}] \right) \tag{26}$$

$$P^{(k)} = \text{softmax} \left( E^{(k)T} w \right) \tag{27}$$

where  $W_R \in \mathbb{R}^{d_e \times d_e}$  are the parameters to learn,  $FFN(\cdot)$  is a feedforward layer and  $E^{(k)T}$  is a matrix in which each column vector is the embedding of an entity at the  $k$ -th step. The Kullback–Leibler divergence [46] measures the difference between two distributions. The loss on one instance is defined as follows:

$$\mathcal{L} = D_{KL}(P', P) \tag{28}$$

where  $P$  denotes the final entity distribution for the forward reasoning process, and  $P'$  denotes the ground truth entity distribution. The loss function of the NSM model trained with strongly supervised data is defined as follows:

$$\mathcal{L}_s^{NSM} = \sum_{k=1}^N D_{KL}(P'^k, P^k) \tag{29}$$

where  $N$  indicates that each instance requires  $N$  reasoning steps.  $P^k$  denotes the entity distribution of the reasoning process at the  $k_{th}$ -step, and  $P'^k$  denotes the ground truth entity distribution at the  $k_{th}$ -step.

#### 4.4 Model pretraining

We pretrain the KBQA models on the constructed dataset *CoSuQue* with strongly supervised labels, so that the model can take advantage of the complete supervised labels to accurately parse the complex semantics of the questions and reason on the question-specific subgraphs.

Since the *CoSuQue* dataset includes a variety of questions with varying degrees of semantic difficulty, we extract an equal amount of data from each type of question to form a new subset, referred to as 'All type' dataset. The parameters of the model are saved to a.ckpt file after the GraftNet model and the NSM model converge on the All type dataset. For the public datasets, we load the saved model parameters before training, and test the model performance after training.

### 5 Experiments

In this section, we utilize the *CoSuQue* dataset to evaluate the impact of strongly supervised labels on the training process and analyse the preference of IR-based models for different types of questions. We pretrain the GraftNet and NSM models on the *CoSuQue* dataset, obtaining the GraftNet<sub>CoSuQue</sub> and NSM<sub>CoSuQue</sub> model, which are then evaluated on the CWQ and WebQSP datasets.

#### 5.1 Dataset construction and implementation details

According to in Sects. 4.1 and 4.2, a variety of complex questions and their corresponding answers can be obtained. Following the traditional architecture of the KBQA model, we retrieve the corresponding subgraphs (multiple triples) from the knowledge base (Freebase)<sup>2</sup> according to the topic entities in the question.

As shown in Figs. 5 and 6, we calculate the statistics of each type of dataset, including the average answer coverage and the average number of triples in the subgraph. The average number of triples indicates the average number of triples  $\langle h, r, t \rangle$  contained in the subgraphs of 30,000 questions. The answer coverage represents the percentage of subgraphs that contain answers to the questions.

To demonstrate the effectiveness of the proposed method, the hyperparameter settings of the two models are consistent with those of Sun et al. [24] and He et al. [19].

**GraftNet.** The number of layers is set to 3 ( $L = 3$ ), the batch size is set to 10, and the learning rate is set to 0.001.

<sup>2</sup> The Knowledge base can be downloaded from <https://developers.google.com/freebase/>.

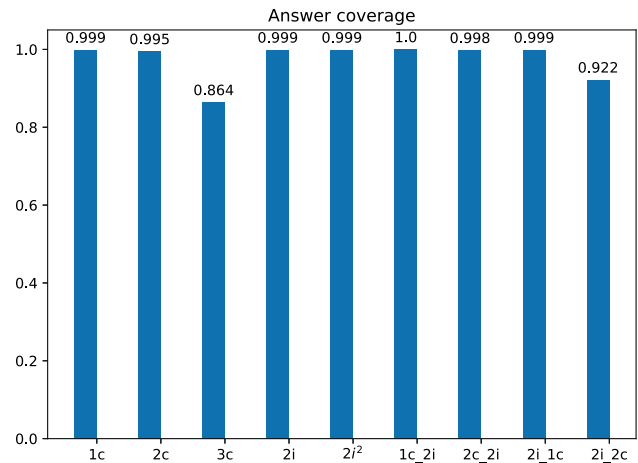


Fig. 5 Answer coverage

The embedding size of both question words is set to  $d = 100$ , and the hidden dimensions of the relations and entities in the knowledge base are set to  $d_r = 100$  and  $d_e = 50$ .

**NSM.** The number of reasoning steps is set to 4 ( $n = 4$ ), the batch size is set to 40, and the learning rate is set to 0.00005. The embedding size of both question words is set to  $d = 300$ , and the hidden dimensions of the relations and entities in the knowledge base are set to  $d_r = 100$  and  $d_e = 50$ .

We optimize the two models with the Adam optimizer, and the number of training epochs set to 100.

#### 5.2 Datasets

**CoSuQue** The *CoSuQue* dataset contains ten subdatasets, including nine types of question subdatasets and one subdataset composed of multiple types of questions mixed in the same proportion, which are denoted as  $\{1c, 2c, 3c, 2i, 2i^2, 1c_{2i}, 2c_{2i}, 2i_{1c}, 2i_{2c}, \text{and } Alltype\}$ . Each subdataset includes 30,000 questions and is divided into a training set, a validation set and a test set according to a weight ratio of 8:1:1.

**WebQuestionsSP (WebQSP)** The WebQSP dataset is one of the commonly used datasets for CQA tasks. This dataset includes 4737 natural language questions that can be answered using the Freebase knowledge base.

**Complex WebQuestions 1.1 (CWQ)** The CWQ dataset contains 34,689 questions that can be answered using the Freebase knowledge base. This dataset was generated from the WebQuestionsSP dataset by extending the question entities or adding constraints to the answers; thus, the CWQ dataset has a significantly higher proportion of complex questions with multi-hop relations and constraint operations.

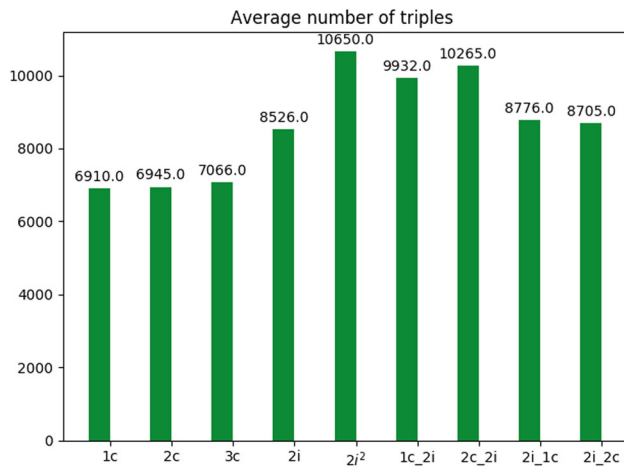


Fig. 6 Average number of triples

### 5.3 Baselines

We consider the following methods in our performance comparison.

**KV-Mem** [47] model maintains a memory table for retrieval, that stores KB facts as key-value pairs, with the head entity and relation stored in the key slot and the tail entity stored in the value slot.

**PullNet** [48] builds on GraftNet and pays more attention to utilizing an iterative process to construct question-specific subgraphs.

**QGG** [28] is an improved staged query graph generation method that constrains the search space by incorporating constraints into the query graph early and using beam search.

**NSM<sub>n</sub>** [19] is a series of ablation models with a teacher-student architecture proposed by He et al.  $NSM_{+f}$  and  $NSM_{+b}$  indicate that the model only uses forwards or backwards reasoning, respectively.  $NSM_{+p}$  and  $NSM_{+h}$  indicate that the model uses parallel or hybrid reasoning, respectively.  $NSM_{+n,-c}$  indicates that the model removes the correspondence loss.

**SGREADER** [49] uses two sources of information, text and a knowledge base, and employs graph attention to achieve effective reasoning.

**EmbedKGQA** [40] employs Roberta to encode complex questions as relation vectors  $e_q$  and uses the ComplEx score to determine the answer entity.

**Tree2Seq** [22] adopts an encoder-decoder framework that encodes the order of the entities and relationships into its representation. The decoder decodes the candidate query into the given question, and the decoding probability is used to select the best query.

**BERT** [33] encodes questions and paths by using the representations of special characters [CLS] for classification.  $BERT_n$  are loaded with BERT model parameters that are pretrained on other binary classification tasks before training.

**KRN** [50] interactively calculates the similarity between question and relation representations through a Siamese network and outputs the highest-scoring path to query the answers in the knowledge base.

**SF-ANN** [51] uses a novel attention mechanism to capture intrinsic dependencies between questions and candidate answers by deeply coupling the complex interactive information between them.

### 5.4 Main results

Following the standard assessment metrics for question answering over knowledge bases tasks, we evaluate the accuracy of the approaches by using the F1 and Hits@1 (H1) metrics. Specifically, Hits@1 indicates whether the answer with the highest score is the correct answer. If the top answer predicted by the model is the correct answer,  $Hits@1 = 1$ ; otherwise,  $Hits@1 = 0$ .

Tables 3 and 4 report the evaluation results of the NSM and GraftNet models on the *CoSuQue* dataset. Table 3 shows the results of the models on different types of complex questions involving only a single type of logical operation. When the models are trained with only the last supervised labels, the resulting models are abbreviated as  $GraftNet_{weakly}$  and  $NSM_{weakly}$ . Table 4 shows the results of the models on complex questions that combine different types of logical operations. When the two models consider questions that require only one type of logical operation, the questions that require interaction operations are often more difficult than those that require chain operations. The experimental results show that the structures of the two models may need to be improved for the interaction operations. In the majority of cases, the performance of the model gradually decreases as the semantic difficulty of the complex question increases. Among the three types of datasets *1c*, *2c*, and *3c*, the model exhibits the worst performance on the *3c* dataset. The results in Tables 3 and 4 show that, in most types, the models trained with complete supervised labels outperformed the models trained with only the last supervised labels.

Table 5 reports the performance of different methods on the CWQ and WebQSP datasets.  $GraftNet_{+PE}$  represents a model that used the embeddings of the entities and relations in the knowledge base that were pretrained by two methods. In this paper, we mainly compare  $GraftNet_{CoSuQue}$  and GraftNet,  $NSM_{CoSuQue}$  and NSM. The  $GraftNet_{CoSuQue}$

**Table 3** Performance comparison of models on different types of complex questions involving one type of logical operation

Model	1c		2c		3c		2i		2i <sup>2</sup>	
	F1	H1	F1	H1	F1	H1	F1	H1	F1	H1
NSM	<b>97.56</b>	<b>97.75</b>	<b>86.65</b>	<b>88.61</b>	<b>64.27</b>	<b>69.97</b>	80.22	84.61	<b>82.45</b>	<b>82.98</b>
NSM <sub>weakly</sub>	96.78	96.37	85.66	87.62	63.05	68.18	<b>82.28</b>	<b>85.31</b>	80.12	81.66
GraftNet	<b>96.03</b>	<b>96.87</b>	<b>79.96</b>	<b>83.99</b>	<b>49.93</b>	<b>60.13</b>	65.16	77.10	<b>88.16</b>	<b>89.29</b>
GraftNet <sub>weakly</sub>	95.34	96.72	79.26	83.61	45.91	56.62	<b>76.56</b>	<b>81.81</b>	86.05	87.36

Bold values indicated by best results

**Table 4** Performance comparison of models on different types of complex questions combining different types of logical operations

Model	1c2i		2c2iasis>		2i1c		2i2c		Alltype	
	F1	H1	F1	H1	F1	H1	F1	H1	F1	H1
NSM	78.66	81.42	<b>73.69</b>	<b>74.80</b>	<b>63.38</b>	<b>68.34</b>	<b>63.40</b>	<b>72.04</b>	<b>68.29</b>	<b>75.32</b>
NSM <sub>weakly</sub>	<b>81.91</b>	<b>84.22</b>	71.81	74.50	54.46	58.32	60.57	70.82	65.57	73.45
GraftNet	82.10	83.66	78.76	81.90	<b>71.29</b>	<b>78.67</b>	<b>50.31</b>	<b>63.03</b>	<b>48.45</b>	<b>55.76</b>
GraftNet <sub>weakly</sub>	<b>82.62</b>	<b>85.15</b>	<b>81.43</b>	<b>82.13</b>	70.35	76.82	49.01	59.31	41.76	48.66

Bold values indicated by best results

**Table 5** Experimental results obtained on the WebQSP and CWQ datasets

	WebQSP		CWQ	
	F1	H1	F1	H1
KV-Mem	38.6	46.7	–	21.1
GraftNet <sub>+PE</sub>	62.4	66.7	–	32.8
PullNet	–	68.1	–	45.9
QGG	72.2	71.9	40.4	44.1
NSM <sub>+f</sub>	64.7	70.7	41.5	47.2
NSM <sub>+b</sub>	65.4	71.1	42.7	47.1
NSM <sub>+p,-c</sub>	66.5	72.5	42.7	47.7
NSM <sub>+h,-c</sub>	66.9	73.0	42.1	47.5
NSM <sub>+p</sub>	66.2	73.9	<b>44.0</b>	48.3
NSM <sub>+h</sub>	<b>67.4</b>	<b>74.3</b>	<b>44.0</b>	48.8
SGREADER	58.0	66.5	–	–
EmbedKGQA	–	66.6	–	–
BERT	63.4	71.2	–	–
BERT <sub>rr</sub>	61.8	71.7	–	–
BERT <sub>re,rm,rr</sub>	62.5	72.3	–	–
Tree2Seq	–	63.9	–	–
KRN	–	64.9	–	–
SF-ANN	59.2	58.1	–	–
GraftNet	45.5	55.5	31.0	33.9
NSM	61.6	66.8	42.4	47.6
NSM (epoch=200)	62.8	68.7	–	–
GraftNet <sub>CoSuQue</sub>	53.3	60.5	33.6	39.0
NSM <sub>CoSuQue</sub>	65.0	<b>73.6</b>	43.0	<b>49.1</b>
NSM <sub>CoSuQue</sub> (epoch=200)	<b>65.1</b>	73.4	–	–

Bold values indicated by best results

model pretrained with CoSuQue outperforms GraftNet on all metrics on the CWQ and WebQSP datasets. The NSM<sub>CoSuQue</sub> model pretrained with CoSuQue achieves better results than the above methods in terms of the Hits@1 metric on the CWQ dataset, which represents the best ability of the model to predict the best answer. The Hits@1 value of the NSM<sub>CoSuQue</sub> model on the WebQSP dataset is higher than that of the NSM model 6.8 trained for 100 epochs, and that of the NSM model 4.9 trained for 200 epochs.

The model inference performance with incomplete knowledge graphs is also important, and we conduct experiments using the sparse WebQSP dataset constructed by Sun et al. [24], which was constructed by downsampling the number of KB facts to 10%, 30%, and 50% of the original data. The experimental results are shown in Table 6 demonstrating that the performance of the GraftNet<sub>CoSuQue</sub> model on knowledge graphs with different degrees of sparseness is better than that of the GraftNet model. The above experimental results demonstrate the effectiveness of our proposed method.

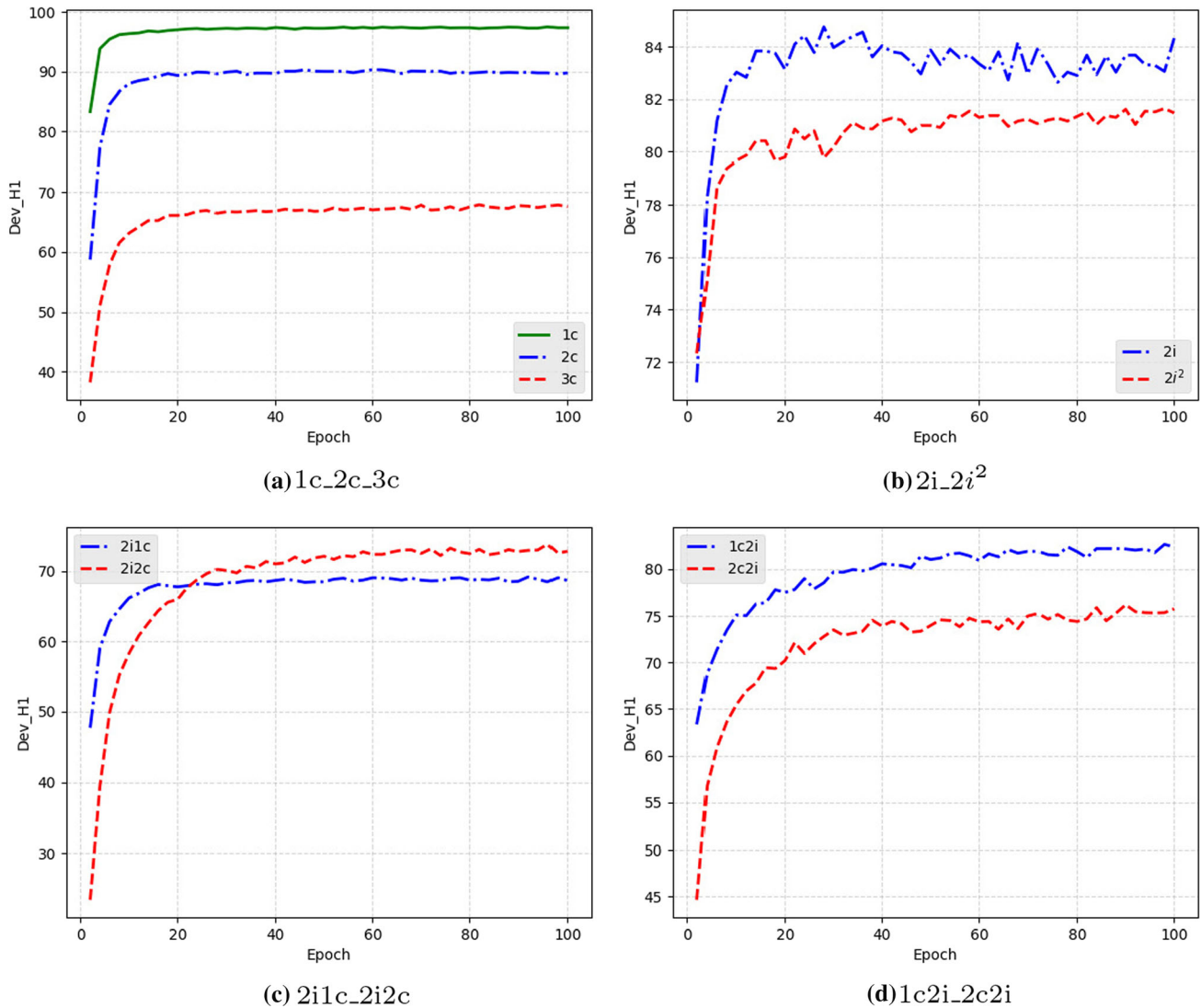
## 5.5 Analysis and visualization

In this section, we conduct a series of visual analysis on the training processes of the GraftNet and NSM models under different supervised labels to address different types of questions, then analyse the model inference process accordingly.

**Table 6** Performance comparison with incomplete KBs of the WebQSP dataset

	10%KB		30%KB		50%KB		100%KB	
	F1	H1	F1	H1	F1	H1	F1	H1
GraftNet	3.97	13.02	10.14	24.66	21.99	37.09	45.52	55.51
GraftNet <sub>CoSuQue</sub>	<b>4.45</b>	<b>15.83</b>	<b>14.00</b>	<b>27.63</b>	<b>25.05</b>	<b>40.36</b>	<b>53.29</b>	<b>60.52</b>

Bold values indicated by best results



**Fig. 7** The training process of the NSM model on different types of question datasets

**5.5.1 Preference for different types of questions**

We visualize the training process of the model for different types of question data. In Fig. 7, the *x*-axis of each sub-graph is the number of epochs, and the *y*-axis is the Hits@1 value of the model on the validation set.

As shown in Fig. 7a, the NSM model converges the fastest on the 1c dataset and has the best performance. The

difficulty of the questions in the 3c dataset relative to 2c dataset is significantly greater than that of 2c relative to 1c. This result demonstrates that for chain-type questions, the semantic difficulty does not increase linearly with the number of reasoning steps.



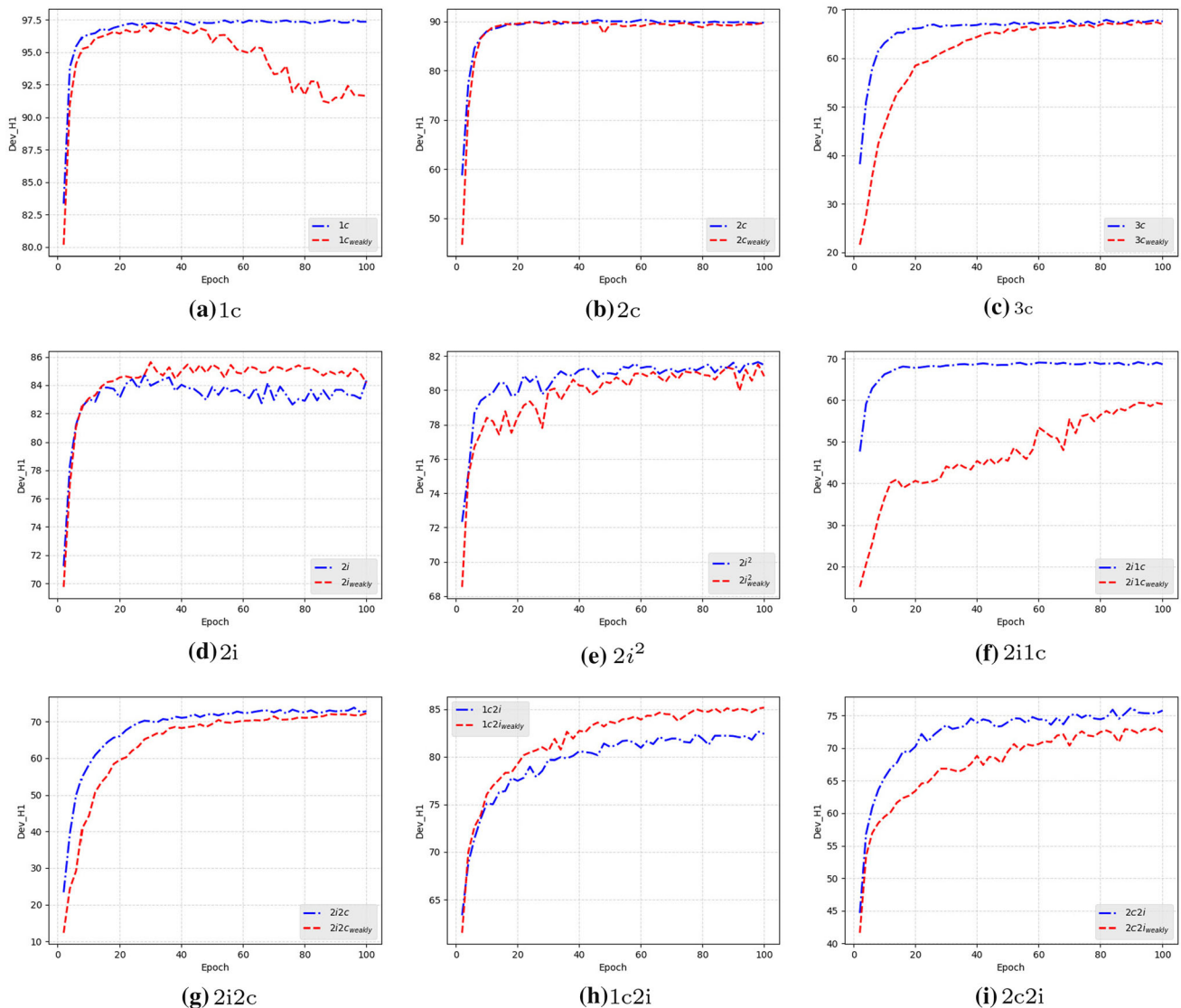
### 5.5.2 Impact of strongly supervised labels

To explore the impact of intermediate supervised labels on model training, we visualized the training process for each type of question with and without intermediate supervised labels. The model shown in Figs. 8 and 9 that was trained with only the last supervised labels is denoted as the ‘ $Type_{weakly}$ ’.

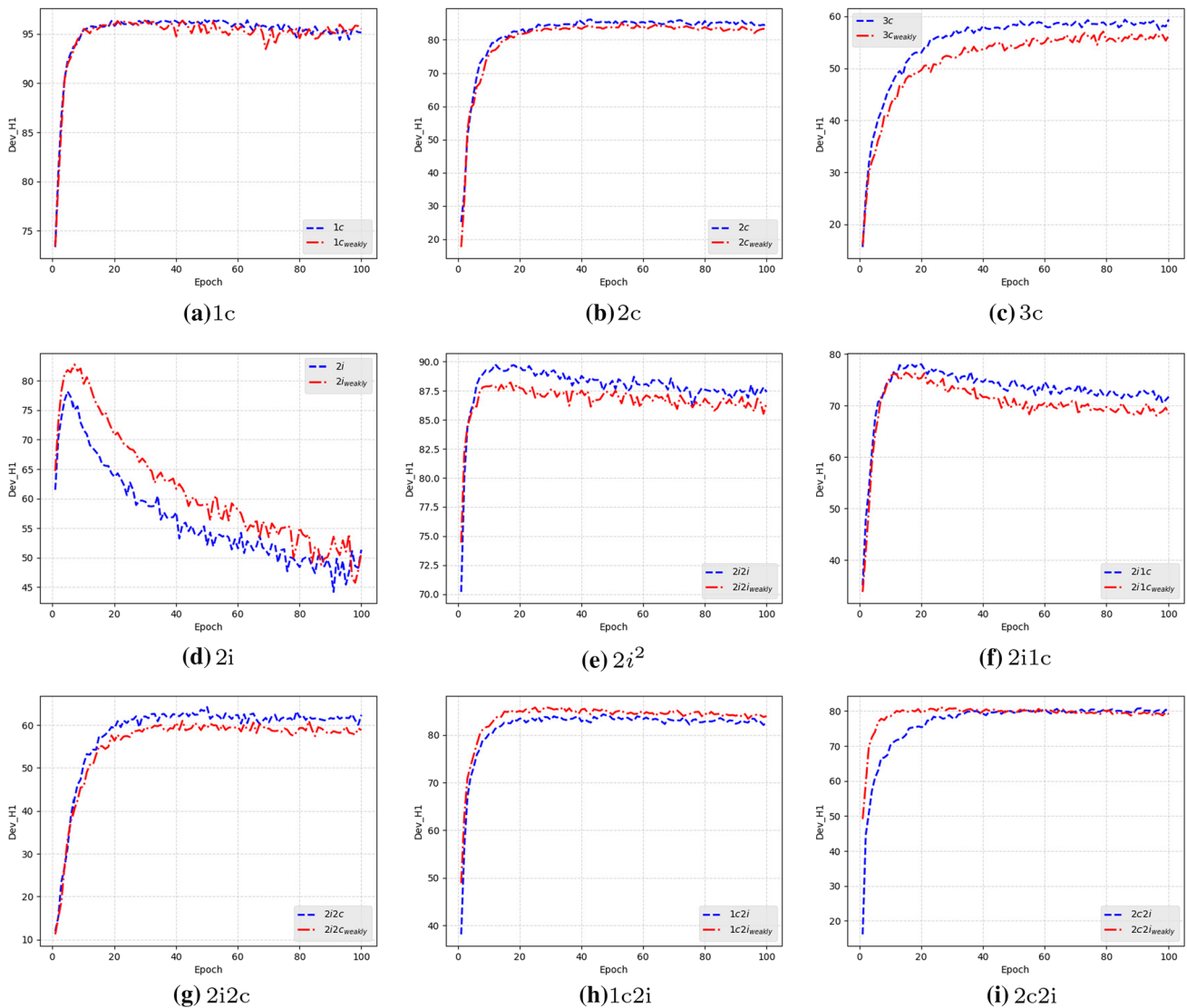
As shown in Figs. 8a and 9a, for the simplest chain questions, such as those in  $1c$ , the training process is more stable when each reasoning step is supervised. Except for  $2i$  and  $1c2i$ , the models using intermediate supervised labels outperformed the models using only the last reasoning step supervised labels on other types of datasets. Comparing the training process in Figs. 8 and 9, the figures demonstrate that when dealing with a certain type of

questions, strongly supervised labels are more helpful for improving the performance of the NSM model, while improvement in the GraftNet model is relatively smaller. This result may have occurred because the GraftNet model contains a mechanism that continuously updates the question representations during the reasoning steps, while the N-step instructions in the NSM model are generated before reasoning.

When GraftNet deal with certain types of question, its performance is better than that of NSM on some types of questions, and when dealing with data mixed with multiple types of questions, the performance of GraftNet is significantly lower than that of NSM. The GraftNet model may not be as good at learning knowledge from various types of questions, as confirmed by Fig. 10a and b. The strongly supervised labels have a greater impact on GraftNet with



**Fig. 8** The impact of different supervised labels on NSM model training



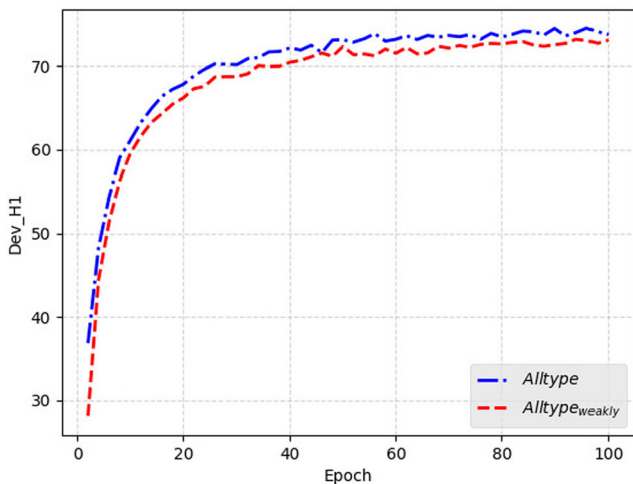
**Fig. 9** The impact of different supervised labels on GraftNet model training

the All type subdataset than the other subdatasets, while this effect is not as obvious with the NSM model. This finding may be due to the fact that the NSM model is more capable of gradually learning knowledge through different types of questions than the GraftNet model. The All type dataset contains 9 types of questions in equal proportions, and for questions containing the same type of logical operations, a question with fewer reasoning steps can be regarded as an intermediate reasoning process of a question requiring more reasoning steps. For example,  $1c$  can be regarded as an intermediate reasoning process of  $2c$ , and  $1c$  and  $2c$  can be regarded as intermediate reasoning processes of  $3c$ .

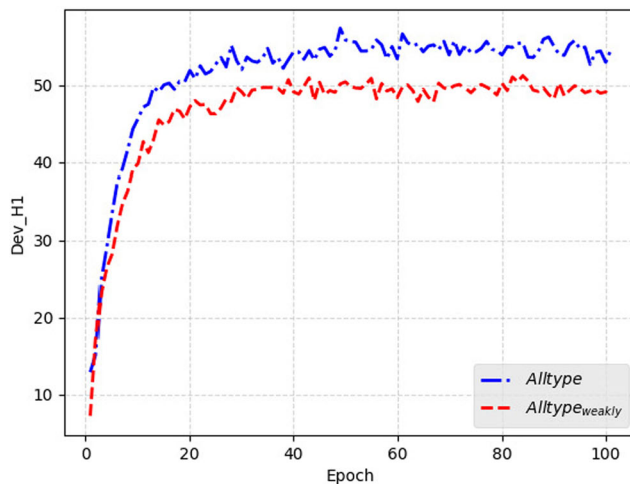
### 5.5.3 Model comparison on the public dataset

Figure 11a and b shows the training processes of the two models on the CWQ and WebQSP datasets. The  $NSM_{CoSuQue}$  and  $GraftNet_{CoSuQue}$  models converged significantly faster than the NSM and GraftNet models. As shown in Fig. 11a, the pretrained model significantly outperforms the model that was not pretrained on *CoSuQue*. After epoch 2, the Hits@1 value of the  $NSM_{CoSuQue}$  reaches 40.81, while the Hits@1 value of the NSM model is 26.91 in the same training epoch.

Figure 12 shows the training process of the NSM model on the WebQSP dataset with a maximum of 200 epochs. When the maximum number of epochs is 100, the  $NSM_{CoSuQue}$  model performs better than the NSM model on both the validation and test sets. When the maximum number of epochs is 200, the  $NSM_{CoSuQue}$  model performs

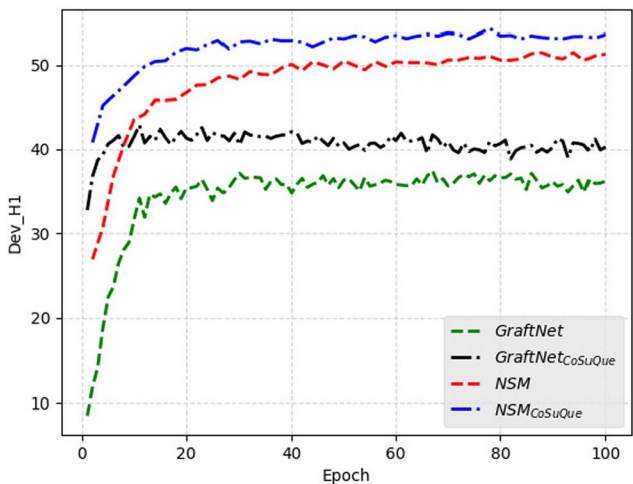


(a) The training process of the NSM model on the All type subdataset with different supervised labels

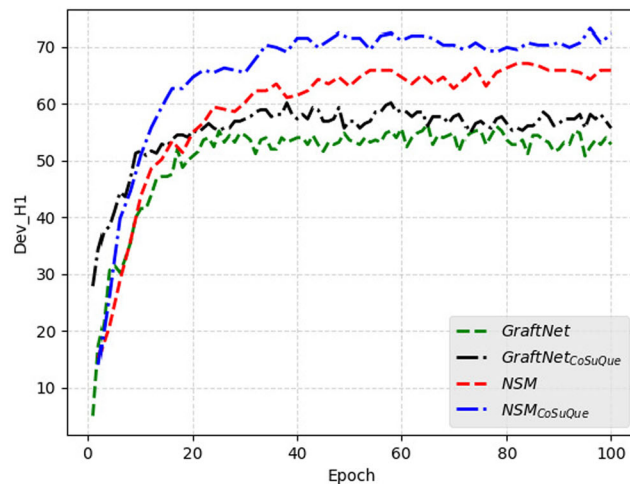


(b) The training process of the GraftNet model on the All type subdataset with different supervised labels

Fig. 10 The training processes of the NSM and GraftNet models



(a) Comparison of model performance on CWQ dataset



(b) Comparison of model performance on WebQSP dataset

Fig. 11 The training process of the two models on public datasets

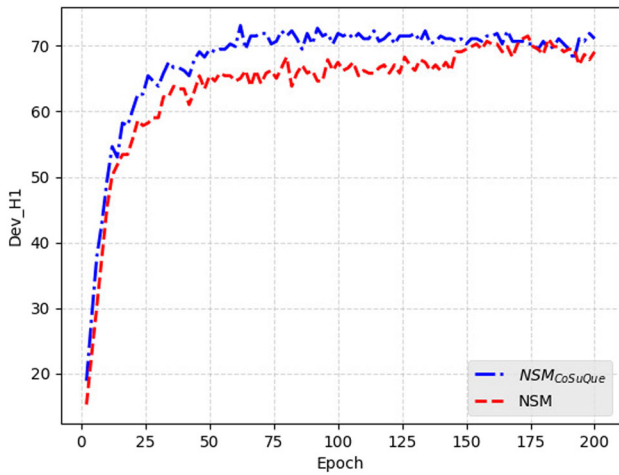
comparably to the NSM model on the validation set. However, on the test set, the Hits@1 value of the NSM<sub>CoSuQue</sub> model is still higher than that of the NSM model. Thus, the experimental results show that the model pretrained on the *CoSuQue* dataset may be more generalizable.

### 5.5.4 Case study

The experimental results show that the strongly supervised labels of the *CoSuQue* dataset can accelerate the model convergence and improve the performance, and the performance of the model on the public dataset is also

improved after the model is pretrained on the *CoSuQue* dataset. We present a case study to illustrate how the *CoSuQue* dataset helps models.

We select examples from the *CoSuQue* dataset to analyse the contribution of strongly supervised labels. Figure 13 shows the different parts of the question that the NSM model focuses on at each reasoning step, with darker squares indicating that the model focuses more on the corresponding word. From the perspective of human reasoning, a model should focus first on the “nationality of Albert Einstein” part of the question and then on the “what are time zones of” part when answering the complex question “what are time zones of nationality of Albert



**Fig. 12** Comparison of model performance on WebQSP dataset (epoch=200)

Einstein?”. As shown in Fig. 13a, the NSM<sub>weakly</sub> model focuses on the same part in the first two steps of reasoning and gradually shifts its attention to the other part in the last two steps of reasoning. Only one keyword, “of”, received high attention in the last reasoning step. As shown in Fig. 13b, the NSM model under the strong supervision condition focuses on different parts of the question in the first and second reasoning step and quickly focuses on the two keywords “of”.

To verify the effectiveness of the method of pretraining the model through the *CoSuQue* dataset, we selected two examples from the CWQ dataset to compare the differences between the NSM model and the NSM<sub>CoSuQue</sub> model.

**Fig. 13** A case from the CoSuQue dataset

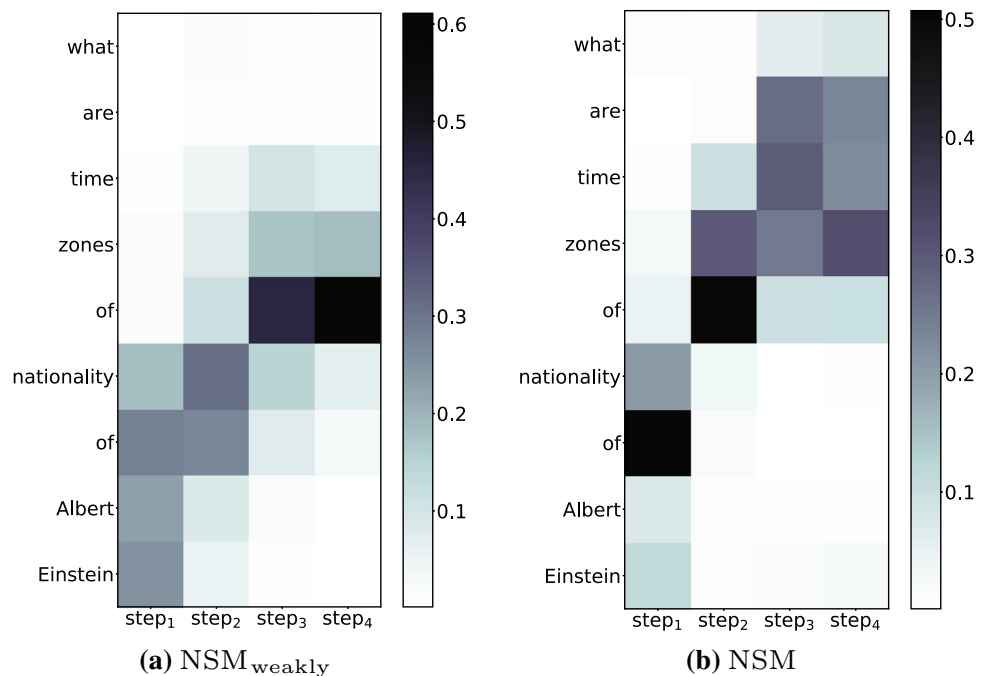


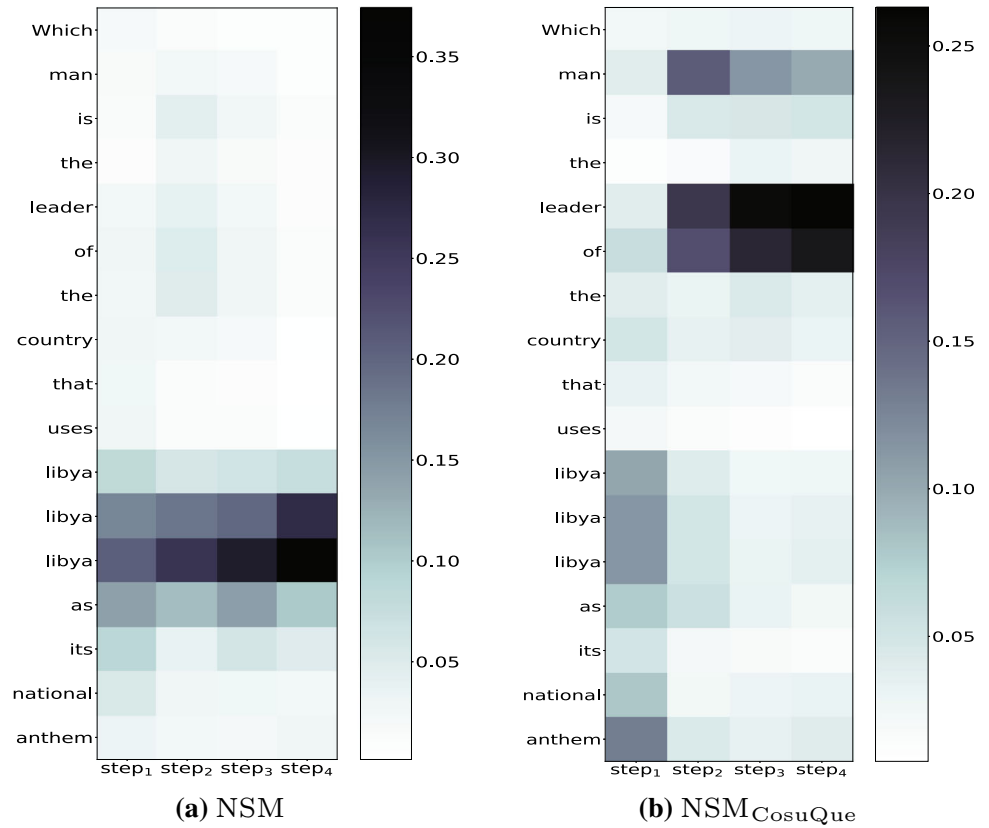
Figure 14a shows the *chain* type of questions in the CWQ dataset. It can be found that the NSM model without pre-trained focuses only on the same part of the question in the four reasoning steps, while the NSM<sub>CoSuQue</sub> (shown in Fig. 14b) model focuses on the other part of the question in the second reasoning step. The reasoning process of the *intersection* type question is different from that of the *chain* type question, and models should address every constraint of the question in the whole reasoning process. The example in Fig. 15 is the *intersection* type question in the CWQ dataset. As shown in Fig. 15a, the NSM model without pretraining focuses only on one question constraint, while the NSM<sub>CoSuQue</sub> model (shown in Fig. 15b) focuses on every constraint in a balanced way.

**5.5.5 Explanation of the model reasoning process**

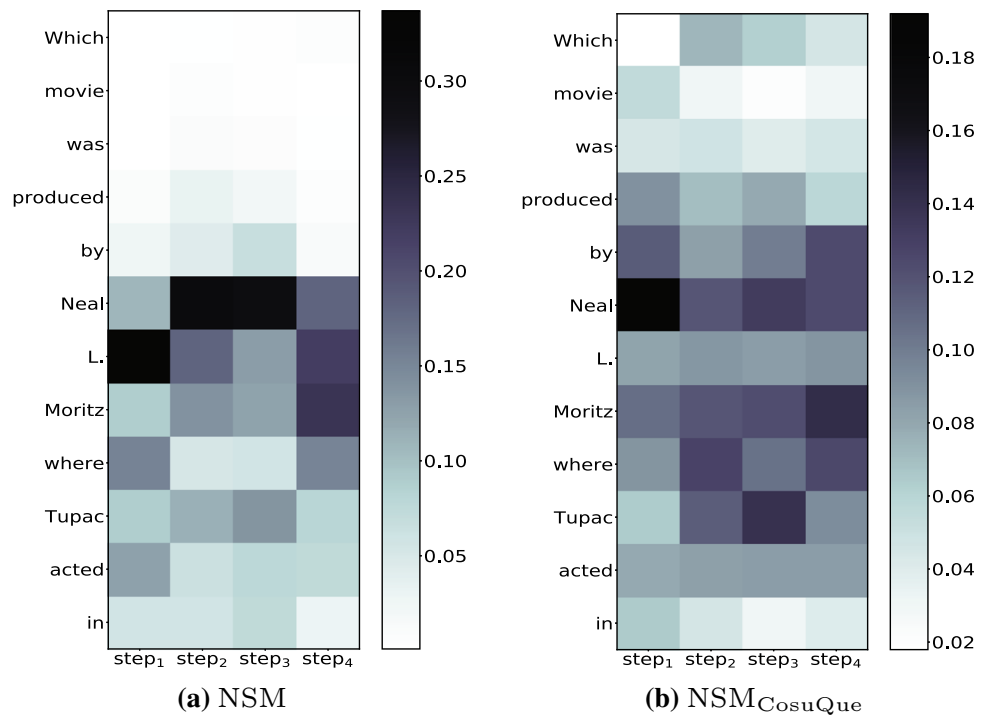
We conducted an in-depth analysis of the NSM and GraftNet models through the *CoSuQue* dataset and explained the reasoning processes of the two models by analysing the experimental results, which can be summarized in the following three aspects.

Comparing the model performance when handling *1c*, *2c* and *3c* types of questions, it can be found that when the number of steps required for complex question reasoning increases, the difficulty of the model in answering questions does not increase linearly. The experimental results of models dealing with questions involving chain logic operations or interaction logic operations prove that questions that require interaction operations are typically more difficult than questions that require only chain operations.

**Fig. 14** A case (*chain* type question) from the CWQ dataset



**Fig. 15** A case (*intersection* type question) from the CWQ dataset



When the GraftNet model is dealing with a single type of complex question, the use of strongly supervised labels improves the performance of the model; however, the

model improvement is less than the improvement of the NSM model with strongly supervised labels, which proves that the algorithm that updates complex question



representations in a step-by-step manner is beneficial for weakly supervised training. Thus, the question representation stepwise update algorithm should be considered when designing models in future work.

In the model training process, each reasoning step has a corresponding supervised label, improving the model's inference ability for most types of complex questions. To a certain extent, it is proven that the reasoning process of these models is similar to that of humans.

## 6 Conclusion and future work

In this paper, we propose a new dataset construction approach and a complex question answering dataset that contains complete supervised labels, and the question types are clearly divided. The proposed method is low-cost, efficient and accurate. In summary, (1) the constructed data could be used to evaluate the performance and preferences of deep learning models at a fine-grained level, which could be helpful for improving models in a more targeted manner. (2) Intermediate supervised labels in datasets could improve the robustness of model training and the model performance on most types of question datasets. (3) Models pretrained on the *CoSuQue* dataset perform better on other datasets.

In future work, we plan to study the following directions: (1) add more types of logic operations, such as negation logic operations, when constructing query graphs to enrich the types of complex questions in the dataset; (2) use deep learning models to transform query graphs into natural language questions to make the question expression more fluent; (3) transform the generated question and query graph into a vector space and explore using the query graph to guide model reasoning.

**Acknowledgements** This research was supported by the Fundamental Research Funds for the Central Universities (Grant number 2020YJS012) and National Key R &D Program of China (No.2018YFC0832300; No.2018YFC0832303).

**Author contributions** XC, YZ and BS designed the study; XC performed the experiments, analysed the data, and wrote the manuscript.

**Funding** This research was funded by the Fundamental Research Funds for the Central Universities (Grant Number 2020YJS012) and National Key R &D Program of China (No.2018YFC0832300; No.2018YFC0832303).

**Availability of data and materials** The datasets generated during and/or analysed during the current study are available from the corresponding author on reasonable request.

**Code availability** Some or models, or code that support the findings of this study are available from the corresponding author upon reasonable request.

## Declarations

**Conflict of interest** The authors have no competing interest to declare that are relevant to the content of this article.

**Ethical approval** Not applicable.

**Consent to participate** Not applicable.

**Consent for publication** Not applicable.

## References

- Jiang Y, Bansal M (2019) Self-assembling modular networks for interpretable multi-hop 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), pp 4474–4484
- Cao X, Liu Y (2021) Coarse-grained decomposition and fine-grained interaction for multi-hop question answering. *J Intell Inform Syst* 58:21–41
- Jiang Y, Bansal M (2019) Avoiding reasoning shortcuts: adversarial evaluation, training, and model development for multi-hop QA. In: Proceedings of the 57th annual meeting of the association for computational linguistics, pp 2726–2736
- Yang Z, Qi P, Zhang S, Bengio Y, Cohen W, Salakhutdinov R, Manning CD (2018) Hotpotqa: a dataset for diverse, explainable multi-hop question answering. In: Proceedings of the 2018 conference on empirical methods in natural language processing, pp 2369–2380
- Cao X, Liu Y, Hu B, Zhang Y (2021) Dual-channel reasoning model for complex question answering. *Complexity* 2021:7367181. <https://doi.org/10.1155/2021/7367181>
- Ren H, Dai H, Dai B, Chen X, Yasunaga M, Sun H, Schuurmans D, Leskovec J, Zhou D (2021) Lego: latent execution-guided reasoning for multi-hop question answering on knowledge graphs. In: International conference on machine learning, pp 8959–8970. PMLR
- Saxena A, Chakrabarti S, Talukdar P (2021) Question answering over temporal knowledge graphs. In: Proceedings of the 59th annual meeting of the association for computational linguistics and the 11th international joint conference on natural language processing (Volume 1: Long Papers), pp 6663–6676
- Kapanipathi P, Abdelaziz I, Ravishankar S, Roukos S, Gray A, Astudillo RF, Chang M, Cornelio C, Dana S, Fokoue-Nkoutche A et al (2021) Leveraging abstract meaning representation for knowledge base question answering. In: Findings of the association for computational linguistics: ACL-IJCNLP 2021, pp 3884–3894
- Gu Y, Kase S, Vanni M, Sadler B, Liang P, Yan X, Su Y (2021) Beyond iid: three levels of generalization for question answering on knowledge bases. In: Proceedings of the web conference 2021, pp 3477–3488
- Xu K, Lai Y, Feng Y, Wang Z (2019) Enhancing key-value memory neural networks for knowledge based question answering. 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), pp 2937–2947
- Auer S, Bizer C, Kobilarov G, Lehmann J, Cyganiak R, Ives Z (2007) Dbpedia: a nucleus for a web of open data. In: The Semantic Web, pp 722–735. Springer

12. Suchanek FM, Kasneci G, Weikum G (2007) Yago: a core of semantic knowledge. In: Proceedings of the 16th international conference on world wide web, pp 697–706
13. Bollacker K, Evans C, Paritosh P, Sturge T, Taylor J (2008) Freebase: a collaboratively created graph database for structuring human knowledge. In: Proceedings of the 2008 ACM SIGMOD international conference on management of data, pp. 1247–1250
14. Li X, Zang H, Yu X, Wu H, Zhang Z, Liu J, Wang M (2021) On improving knowledge graph facilitated simple question answering system. *Neural Comput Appl* 33(16):10587–10596
15. Min S, Zhong V, Zettlemoyer L, Hajishirzi H (2019) Multi-hop reading comprehension through question decomposition and rescoring. In: Proceedings of the 57th annual meeting of the association for computational linguistics, pp 6097–6109
16. Liang C, Berant J, Le Q, Forbus K, Lao N (2017) Neural symbolic machines: learning semantic parsers on freebase with weak supervision. In: Proceedings of the 55th annual meeting of the association for computational linguistics (Volume 1: Long Papers), pp 23–33
17. Qiu Y, Wang Y, Jin X, Zhang K (2020) Stepwise reasoning for multi-relation question answering over knowledge graph with weak supervision. In: Proceedings of the 13th international conference on web search and data mining, pp 474–482
18. Qiu Y, Zhang K, Wang Y, Jin X, Bai L, Guan S, Cheng X (2020) Hierarchical query graph generation for complex question answering over knowledge graph. In: Proceedings of the 29th ACM international conference on information & knowledge management, pp 1285–1294
19. He G, Lan Y, Jiang J, Zhao WX, Wen J-R (2021) Improving multi-hop knowledge base question answering by learning intermediate supervision signals. In: Proceedings of the 14th ACM international conference on web search and data mining, pp 553–561
20. Luo K, Lin F, Luo X, Zhu K (2018) Knowledge base question answering via encoding of complex query graphs. In: Proceedings of the 2018 conference on empirical methods in natural language processing, pp 2185–2194
21. Chen Y, Li H, Hua Y, Qi G (2021) Formal query building with query structure prediction for complex question answering over knowledge base. In: Proceedings of the Twenty-Ninth international conference on international joint conferences on artificial intelligence, pp 3751–3758
22. Zhu S, Cheng X, Su S (2020) Knowledge-based question answering by tree-to-sequence learning. *Neurocomputing* 372:64–72
23. Han J, Cheng B, Wang X (2020) Open domain question answering based on text enhanced knowledge graph with hyperedge infusion. In: Findings of the association for computational linguistics: EMNLP 2020, pp 1475–1481
24. Sun H, Dhingra B, Zaheer M, Mazaitis K, Salakhutdinov R, Cohen W (2018) Open domain question answering using early fusion of knowledge bases and text. In: Proceedings of the 2018 conference on empirical methods in natural language processing, pp. 4231–4242
25. Talmor A, Berant J (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), pp. 641–651
26. Yih SW-t, Chang M-W, He X, Gao J (2015) Semantic parsing via staged query graph generation: Question answering with knowledge base. In: Proceedings of the joint conference of the 53rd annual meeting of the ACL and the 7th international joint conference on natural language processing of the AFNLP
27. Hao T, Li X, He Y, Wang FL, Qu Y (2022) Recent progress in leveraging deep learning methods for question answering. *Neural Comput Appl* 34:2765–2783. <https://doi.org/10.1007/s00521-021-06748-3>
28. Lan Y, Jiang J (2020) Query graph generation for answering multi-hop complex questions from knowledge bases. In: Association for computational linguistics
29. Sun Y, Zhang L, Cheng G, Qu Y (2020) Sparqa: skeleton-based semantic parsing for complex questions over knowledge bases. In: Proceedings of the AAAI conference on artificial intelligence, vol. 34, pp 8952–8959
30. Devlin J, Chang M-W, Lee K, Toutanova K (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), pp 4171–4186
31. Das R, Zaheer M, Thai D, Godbole A, Perez E, Lee JY, Tan L, Polymenakos L, McCallum A (2021) Case-based reasoning for natural language queries over knowledge bases. In: Proceedings of the 2021 conference on empirical methods in natural language processing, pp 9594–9611
32. Zhang Y, Dai H, Kozareva Z, Smola AJ, Song L (2018) Variational reasoning for question answering with knowledge graph. In: Thirty-Second AAAI conference on artificial intelligence
33. Yan Y, Li R, Wang S, Zhang H, Daoguang Z, Zhang F, Wu W, Xu W (2021) Large-scale relation learning for question answering over knowledge bases with pre-trained language models. In: Proceedings of the 2021 conference on empirical methods in natural language processing, pp 3653–3660
34. Bordes A, Usunier N, Garcia-Duran A, Weston J, Yakhnenko O (2013) Translating embeddings for modeling multi-relational data. *Adv Neural Inform Process Syst* 26
35. Wang Z, Zhang J, Feng J, Chen Z (2014) Knowledge graph embedding by translating on hyperplanes. In: Proceedings of the AAAI conference on artificial intelligence, vol. 28
36. Lin Y, Liu Z, Sun M, Liu Y, Zhu X (2015) Learning entity and relation embeddings for knowledge graph completion. In: Twenty-ninth AAAI conference on artificial intelligence
37. Ji G, He S, Xu L, Liu K, Zhao J (2015) Knowledge graph embedding via dynamic mapping matrix. In: Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing (Volume 1: Long Papers), pp 687–696
38. Peng Y, Zhang J (2020) Lineare: simple but powerful knowledge graph embedding for link prediction. In: 2020 IEEE international conference on data mining (ICDM), pp 422–431. IEEE
39. Chao L, He J, Wang T, Chu W (2021) Pairre: knowledge graph embeddings via paired relation vectors. In: Proceedings of the 59th annual meeting of the association for computational linguistics and the 11th international joint conference on natural language processing (Volume 1: Long Papers), pp 4360–4369
40. Saxena A, Tripathi A, Talukdar P (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, pp 4498–4507
41. Trouillon T, Welbl J, Riedel S, Gaussier É, Bouchard G (2016) Complex embeddings for simple link prediction. In: International conference on machine learning, pp 2071–2080. PMLR
42. Ren H, Hu W, Leskovec J (2019) Query2box: reasoning over knowledge graphs in vector space using box embeddings. In: International conference on learning representations
43. Liu L, Du B, Ji H, Zhai C, Tong H (2021) Neural-answering logical queries on knowledge graphs. In: Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining, pp 1087–1097

44. Zhang Z, Wang J, Chen J, Ji S, Wu F (2021) Cone: cone embeddings for multi-hop reasoning over knowledge graphs. *Adv Neural Inform Process Syst* 34:19172–19183
45. Haveliwala TH (2003) Topic-sensitive pagerank: a context-sensitive ranking algorithm for web search. *IEEE Trans Knowl Data Eng* 15(4):784–796
46. Kullback S, Leibler RA (1951) On information and sufficiency. *Ann Math Stat* 22(1):79–86
47. Miller AH, Fisch A, Dodge J, Karimi A-H, Bordes A, Weston J (2016) Key-value memory networks for directly reading documents. In: *EMNLP*
48. Sun H, Bedrax-Weiss T, Cohen W (2019) Pullnet: Open domain question answering with iterative retrieval on knowledge bases and text. 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)*, pp 2380–2390
49. Xiong W, Yu M, Chang S, Guo X, Wang WY (2019) Improving question answering over incomplete kbs with knowledge-aware reader. In: *Proceedings of the 57th annual meeting of the association for computational linguistics*, pp 4258–4264
50. Shen Y, Yang M, Li Y, Wang D, Zheng H, Chen D (2021) Knowledge-based reasoning network for relation detection. *IEEE Trans Neural Net Learn Syst*. <https://doi.org/10.1109/TNNLS.2021.3123751>
51. Zhang Y, Jin L, Zhang Z, Li X, Liu Q, Wang H (2022) Sf-ann: leveraging structural features with an attention neural network for candidate fact ranking. *Appl Intell* 52(5):5841–5856

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.